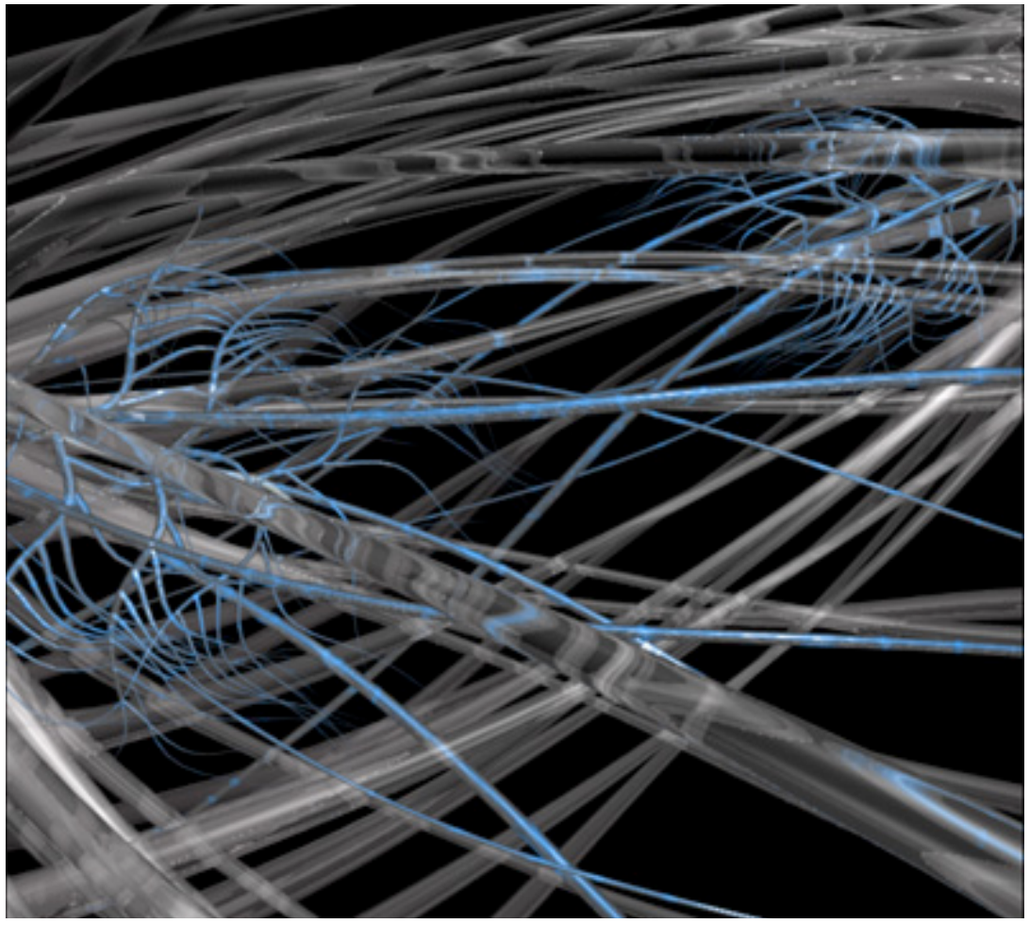
BIOLOGICAL BITS

A BRIEF GUIDE TO THE IDEAS AND ARTEFACTS OF COMPUTATIONAL ARTIFICIAL LIFE

PDF edition

ALAN DORIN



Copyright

Biological Bits: a brief guide to the ideas and artefacts of computational artificial life.
1st PDF edition, Animaland, Melbourne, Australia 2014.

ISBN-10: 0646918710
ISBN-13: 978-0-646-91871-6

To students, educators and supervisors

This is a non-interactive version of the first edition of the interactive book, *Biological Bits* for Apple iBooks. It has been prepared especially for readers without access to interactive book-reading software but lacks the movies and interactive software of the iBook. All text and images from the iBook have been included here, but page numbers may vary between the two versions.

Undergraduate students and educators.

This text can act as preliminary reading for students undertaking semester-long undergraduate courses or projects. I hope the educator might select relevant chapters for students to read prior to lectures or tutorials. The material can serve to prompt students to begin thinking about Artificial Life and explore software based on its principles. Coupled with quiz or discussion questions targeted at the educator's needs, I hope it serves as a useful way to begin to "flip" the classroom.

Postgraduate students and supervisors.

My aim is to provide a broad, engaging text covering the main activities of Artificial Life. I consciously refer to historical work and to what I personally feel to be exciting areas for new research. I feel that the risk of approaching Artificial Life narrowly is real and the result will be an unhealthy compartmentalisation that other disciplines struggle to break. Let's not go there! Some level of exposure across the range of approaches to creating Artificial Life is, I think, likely to be of considerable benefit. If nothing else, an understanding of the material in this text can make attendance at conferences, workshops and symposia on Artificial Life much more rewarding.

Please send me an email to let me know if you use the book in your own research or teaching, or if you have any thoughts on how to improve this work in progress.

Introduction

In its broadest sense, Artificial Life encompasses the creation of imaginary beings and the engineering of life-like artefacts. The artefacts often take the form of fine technology and may act as proxies for nature, allowing us to study or engage with biology through models. With such a wide scope, clearly the practice is interdisciplinary. Besides the biological sciences, anyone who wishes to understand Artificial Life thoroughly would benefit from an investigation of art, technology, social science and the theories, history and philosophy of these fields. But life is short. In reality, everybody comes to the discipline with their own predispositions and interests. The inquisitive will find so much here that there is little chance of escape. Artificial Life arguably addresses the most demanding, challenging, and profound questions that have ever been asked. How did life get here; what is it; and where might it go? By extension, these questions reference *us*. Artificial Life encompasses our deep history, our present and our future.

We are clockwork figures, our springs wound for a short time. We are distinguished from beasts only by our god-given immortal souls that grant us the faculty of Reason.

Are we puppets tugged this way and that in our thoughts and behaviour by threads of gold when we are moral or enlightened, and by baser cords when we are crude, decadent or foolish? Are we the puppets of the gods?

Generally, Artificial Life might be thought of as the abstraction and interpretation of natural biological phenomena in artificial media. But a higher-level perspective is also revealing. We could say that the medium of Artificial Life is “representation”, itself an abstract, intangible concept. Artificial Life’s representations may take on forms as diverse as speculative thoughts conveyed in theoretical and philosophical texts, through fiction, the visual and sonic arts, or as physical ma-

chinery and implemented software. They may even be assembled using nature's own biotic building blocks. But implicit in every artificial life form lies a theory of what life is. As these theories have metamorphosed, diverged and fused over the centuries, what was once a system provoking difficult questions concerning the nature of life may become an oddity in a museum, or spare parts for the next venture. Sometimes, a generation of Artificial Life's engineers dismiss the work of their predecessors as misguided, quaint, or based on a simplistic theory of life. For instance, we no longer think that humans are clockwork machines with a soul, neither do we believe that our brains are digital computers. But these views, and countless others we might be tempted to scoff at, have concerned some of the greatest thinkers of our past. Many of them remain valuable contributions in their own right, flavouring current interpretations of living systems, our language, and our conventions incontrovertibly. Some views are especially precious for the window into history they provide. Hence, when we examine the bits and bytes of computational Artificial Life, it pays to be mindful of two points: that, in time, current theories are as likely to be superseded as the ideas preceding them; and that even concepts specific to digital artificial life emerge from patterns generated in other media. The concept of *life* is quick.

Do we have only an illusion of free-will? Is our life determined by Fate and the whims of gods and monsters?

We are the pinnacle of a god's creation, the goal towards which evolution has been striving.

A.D.
Melbourne, 2014

Hashtag #biolbits on Twitter

General introductory reading

Langton, C. G., Ed. (1995). "Artificial Life, An Overview". Cambridge Massachusetts / London England, MIT Press.

Levy, S. (1992). "Artificial Life, the quest for a new creation". London, Penguin Books.

Riskin, J., Ed. (2007). "Genesis Redux, Essays in the History and Philosophy of Artificial Life". Chicago/London, University of Chicago Press.

About the author

Alan Dorin is associate professor of computer science at Monash University, Melbourne, Australia. His research in artificial life spans ecological and biological simulation, artificial chemistry and biologically-inspired electronic media art. Alan also studies the history of science, technology, art and philosophy especially as these pertain to artificial life.

- alan.dorin@monash.edu
- www.csse.monash.edu.au/~aland
- @NRGBunny1

A note on references and images

I have provided many references for further reading. In some cases I have elected to suggest general information, survey articles, or recent papers that cover obscure or less accessible earlier work. At other times I felt it was more helpful or important to refer to original texts, however historical they might be. In either case, my hope is that the selected references are informative. A quick web-search (e.g. google scholar) should turn most of them up.

Acknowledgments

This book was created in the Faculty of Information Technology, Monash University, Australia. Thank you especially to Steve Kieffer for implementing the special purpose widgets and to Bernd Meyer for supporting this project. Thank you Ned Greene, Ben Porter, Kurt Fleischer, Craig Reynolds, Greg Turk, William Latham, Richard Thomas, Tobias Gleissenberger, Frederic Fol Leymarie, Mani Shrestha, Robert and Sara Dorin, Hiroki Sayama, and Patrick Tresset for allowing me to use your images and animations without which the pages of this book would be primarily Georgia Regular 16. Thanks also to my collaborators and colleagues. This book is derived from the work and discussions I have had with you, especially discussions relating to Artificial Life with (in no particular order) Kevin Korb, Adrian Dyer, John Crossley, Jon McCormack, Taras Kowaliw, Aidan Lane, Justin Martin, Peter McIlwain, John Lattanzio, Rod Berry, Palle Dahlstedt, Troy Innocent, Mark Bedau, Christa Sommerer, Laurent Mignoneau, Erwin Driessens, Maria Verstappen, Mitchell Whitelaw, Nic Geard, Gordon Monro, Barry McMullin, Inman Har-

vey, Takashi Ikegami, Tim Taylor, Ollie Bown, Rob Saunders, Alice Eldridge and Reuben Hoggett. But also others too numerous to name – thanks for your valuable insights over the years. It would be remiss of me to leave out my students, especially those who have put up with me as doctoral, masters or honours supervisor. Thanks for your sharp questions and valuable insights. These have helped immeasurably to shape the contents of this book.

Table of Contents

CHAPTER 1

Technological history

Artificial Life is a discipline that studies biological processes by producing and exploring the properties of models. Many recent Artificial Life models are computer simulations, but Artificial Life is arguably better understood from a broader perspective; it is the invention and study of technological life.

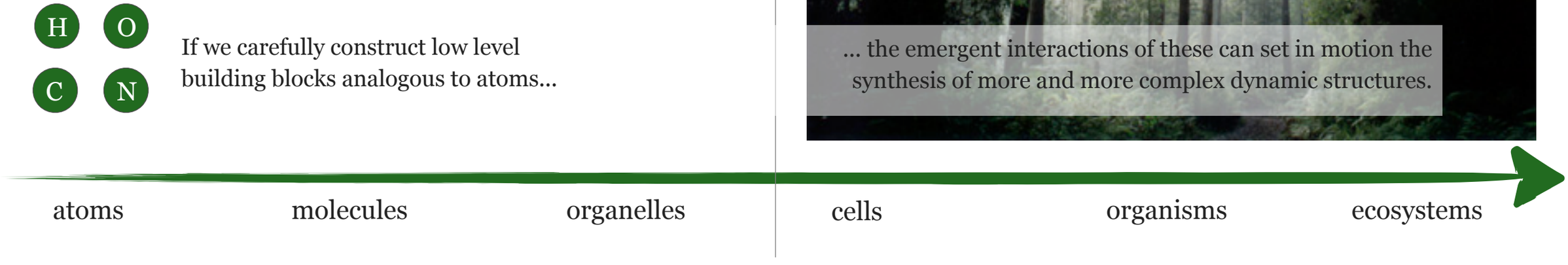
SECTION 1

What is Artificial Life?

TOPICS

1. Emergence and synthesis
2. Artificial life and intelligence
3. What is *life*?

How can we use technology to build living systems? This question has challenged the creativity of humans for millennia. Lately, attempts to answer it have employed computer technology, but this is just the latest trend in a long line of engineering practice that stretches back beyond antiquity. Each attempt has used the technological methods of its era, and each is woven with thread into the tangle of philosophical perspectives that have allowed artificial life's inventors to see their devices as lifelike. In this section we will discuss a couple of themes that underly today's Artificial Life, *synthesis* and *emergence*, and see how they relate to the ideas of Artificial Intelligence and the nature of life.



Emergence and synthesis

Local interactions between simple elements are said to facilitate the *emergence* of complex structures and behaviours at the level of the elements' collection. A commonly cited example of emergence in Artificial Life is that of a flock of birds or school of fish. No individual starling (for example) in a flock keeps track of what all the others are doing. There is no orchestration or management from above. Each bird only examines its neighbours' behaviour and makes a decision about what to do next based on the limited information available to it. The result is, nevertheless, a large, coherent entity – a flock.

The *synthesis* of complex behaviour in this way is one goal of current Artificial Life research. The behaviour is generated from the "bottom up" in the sense that it lacks an overseer who keeps track of the big picture. The individual, and often relatively simple components of the emergent entity, are responsible for its construction. There is no central controller or manager.

A wander through a forest, or a dive into a coral reef, reveals just how complex and diverse nature's bottom up structures can be. Astonishingly, this complexity is built by self-assembly and self-organisation of physical and chemical building blocks into intricate, dynamic hierarchies. There is no plan, and there doesn't need to be one.

The concept of *emergence* is itself subject to philosophical discussion. Is emergence relative to the perspective of the observer? Is there such a thing as "true" emergence whereby an outcome of the interaction of elements cannot, in principle, be predicted from a description of the parts and their relationships? How does emergence relate to the concept of *supervenience*, a term with very similar applications?

Since the creation of the universe sub-atomic particles have assembled themselves into atoms of different types. The atoms self-assemble into molecules, including among them, those essential to life, for instance water. Molecules have properties and exhibit behaviours that are not a characteristic of their atomic components. These properties and behaviours are themselves emergent. For instance, liquid water's excellent action as a solvent is emergent from the interactions of its Hydrogen and Oxygen atoms at 1 (Earth) atmosphere and 20 degrees Celsius.

At some stage in Earth's history, basic molecules appear to have given rise to at least one system capable of self-replication. There are numerous theories about how this might have happened, but the detail of the transition from a "soup" of chemicals to the first populations of replicating molecules remains a mystery. Still, from here, the process of evolution was initiated, first with single cells. Then cells evolved sufficient inter-dependence that communities emerged. Some of these became so tightly knit they became multicellular organisms encapsulated by a single membrane. Many varieties of organism co-operate in the synthesis of marine and terrestrial ecosystems. These are emergent communities of many species, their abiotic environments, and the interactions between all of these components. As well as occupying niches within the different habi-

tats generated by abiotic physical and chemical processes – such as erosion, precipitation, ocean tides, geothermal activity, sunlight and shadow – organisms construct their own habitats via *niche construction*. Their environments are as much a side-effect of life's existence as they are of forces external to organisms.

This level-building process is what is meant by the synthesis of complexity through emergence. These themes are typical of much of the research done in Artificial Life today. How can we get our technology to mimic or reproduce this behaviour? Well, that is an open problem! Many approaches have been tried, each adopts a different kind of basic building block and attempts to show how it can be used to construct something more complex than simply the "sum of its parts".

Artificial life and intelligence

The concept of *intelligence* is at least as slippery as *life*, perhaps more so. The longterm goal of Artificial Intelligence (AI) is to build human-level intelligence and one day move beyond it. On the way to this grand challenge a number of lesser but still exceedingly tricky sub-goals must first be reached. AI involves building machines that solve problems requiring intelligence. Playing chess, devising original mathematical proofs, writing poetry, learning to recognise a Coke can left on a desk, navigating a maze, conducting a conversation with a human... these have all been, at one time or another, suitable candidates for study. The kinds of subproblems AI investigates are concerned with reasoning, knowledge acquisition (learning), its storage or representation and its application. Planning, language use and communication, navigation, object manipulation and game play also fall within AI's scope.

GOFAI is usually "symbolic" in the sense that it involves machine manipulation of symbols according to explicit rules. GOFAI does not encompass all AI research. For instance *non-symbolic AI*, itself dating back well over half a century, uses artificial neural networks, "fuzzy logic" or cybernetic systems to make decisions about how a machine should respond to its environment. These latter approaches overlap with methods frequently employed in Artificial Life.

The traditional approach to realising artificial intelligence is sometimes (facetiously?) dubbed GOFAI, *good old fashioned AI*. GOFAI typically tackles problems from the "top down" by setting a goal and subdividing it into sub-goals. These are repeatedly subdivided into sub-sub-goals, until the achievement of the sub-...-sub-goals is trivial and can be executed by a simple machine or program. Together, the achievement of the sub-goals by the machine results in the solution of the initial big problem. For instance, suppose the big goal is to construct a Lego™ dollhouse. The problem can be solved by subdivision into the need to build a roof, rooms and their contents. The contents include wall fixtures, furnishings and a fireplace. The fireplace might be subdivided into a hearth, mantelpiece, firebox, grate, chimney and chimney pots. And so on. The building tasks get simpler and simpler as the subdivision continues. A similar subdivision of goals might allow a robot to construct a plan in order to travel to the far side of a room. The bottom

level of the subdivision must be simple commands to the robot's locomotory system such as, "turn on the left-front motor at half-speed for one second".

This reductive approach is often contrasted against that of Artificial Life's preference for bottom up synthesis.

One popular activity within contemporary and historical Artificial Life is to (try to) build technological systems that rival the diversity and complexity of individual organism behaviour. For obvious reasons, even if attempted from the bottom up, this research agenda sometimes overlaps with AI's. It is certainly possible to synthesise the equivalent of instinctive creature gaits and reflexes from the bottom up without recourse to the *intelligence* of a robot or other artificial agent. But as soon as the engineered creature is expected to use its gait or respond in a non-reflexive way to environmental stimulus, the project relates to AI's activities. An Artificial Life purist might argue that most creatures respond to their environment in a very reflexive way, taking into account only local conditions and the organism's current internal state. As it bumbles around a complex environment, an organism's interactions naturally appear complex, even though its behavioural machinery is actually quite simple. Such a researcher might forgo any interest in route (or other) planning and goal setting, hallmarks of GOFAI probably not practiced by ants and cockroaches. If the animal under consideration is a primate it is apparent that, under some circumstances at least, a degree of planning and goal setting is necessary to effectively model its behaviour.

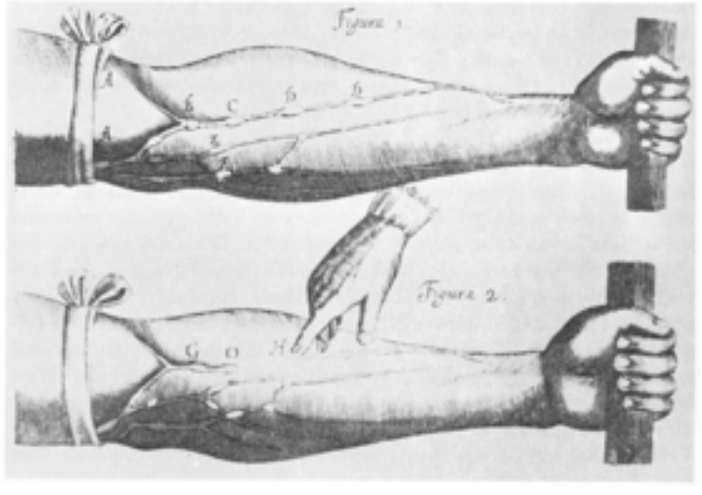
AI research can be considered a subset of Artificial Life, albeit one that many Artificial Life researchers are tempted to skirt around. As far as we know, intelligence only occurs in living systems. That is how it has appeared on Earth anyway. Perhaps one day intelligence might exist independently of life, but I would hazard a guess that this will only be when we, or some other living intelligence, engineer it so. It would be very surprising indeed to find a natural intelligent system that was not also alive.

What is *life*?

The field of Artificial Life is itself emergent from the interactions of its researchers. Hence, there is no leading figure dictating to all Artificial Life researchers that they consider the question, *What is life?* Many in the field are not at all concerned about this issue. To an outsider or newcomer it might seem very surprising that only a few (professional and amateur) philosophers in the field care! But even biologists and ecologists can make a living without having a thorough understanding of the philosophy underpinning the subject of their attention. All the same, many libraries of text produced by the greatest minds in history have addressed the subject. Personally I consider an understanding of this question to be paramount to Artificial Life research, to Biology, to Ecology and even to humanity. It is arguably one of the most fundamental concerns of any conscious life form. Well, it should be. And I count thinkers from Aristotle to Schrödinger among those in agreement. Regardless of its importance, *What is life?*, is an enticing challenge.

Definitions of *life* usually fall into a few distinct categories based on what organisms have, what they do, what they are made of, how they are made, and what they are a part of. That is, most (but not all) definitions are based on the properties of organisms, or on the relationships they have with other things.

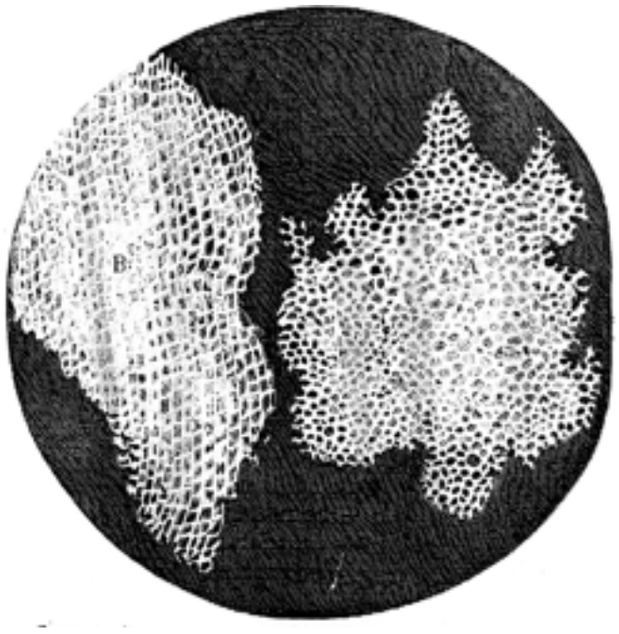
Properties of organisms. Simple and obvious properties organisms have that might allow them to be distinguished from non-life include eyes, limbs, hearts and, some have argued, souls. Aristotle himself was an early proponent of souls. But he didn't have in mind the kind of soul that Christians later adopted. Aristotle's concept of the soul (*psyche* / Ψυχή) came in five flavours. Most basic of all is the nutritive and generative soul possessed by plants and all living things. Next, the sentient soul and the appetitive soul are common to all animal life. Some animals also have locomotive soul that enables them to move. Last of the five, humankind alone has rational soul. Aristotle believed that the types of soul and their associated *vital heat* were distributed around the body as "pneuma"-infused blood. This isn't simply air-infused blood, the reference here is to a "philosophical" substance that drives life.



Part of William Harvey's evidence demonstrating the circulation of the blood could be observed by anybody in 1628 by applying slight pressure to a vein in their arm. Image out of copyright.

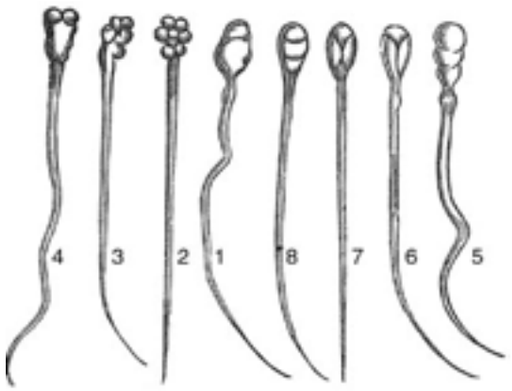
Aristotle had understood that organisms ate and grew. Some could perceive their environment and had drives and goals. Some could move and, humans at least, could reason. About these things he was correct! He didn't yet possess sufficient knowledge to understand *how* these phenomena were generated, but he understood more than most of his contemporaries.

Most improvement in our understanding of life has come about since the 16th century. Why did it take so long? From around 200 CE the ideas of a Roman physician, surgeon and philosopher, Galen of Pergamon, dominated Western medicine and our understanding of life. Galen's ideas drew heavily from those of the legendary 5th C BCE Greek physician Hip-



The illustration by 17th century microscopist Robert Hooke of the cells he discovered in cork.

Hooke, R. (1655). "Micrographia or Some physiological descriptions of minute bodies made by magnifying glasses with observations and inquiries thereupon", London, p.112. Image out of copyright.



Antoni van Leeuwenhoek's illustration of sperm seen through the microscope. 1-4 are human sperm, 5-8 are dogs'. Van Leeuwenhoek was amazed at the complexity of life at such a tiny scale. He was the first to observe the countless microorganisms living out their entire lives in a drop of water. Image out of copyright.

pocrates, as well as Plato and Aristotle. Galen's concept of the soul was close to Plato's. It had three parts: vital spirit for heat and life, desire for nutrition, reproduction and growth, and wisdom for feelings and thoughts. Like Aristotle, Galen thought the stuff was distributed around the body in the blood. The idea that blood (and presumably pneuma with it) might *circulate* within the body, and not be consumed by it, had not yet entered Western medicine. In 1628 English physician William Harvey convincingly argued circulation was correct, but nobody even in this time understood how the blood's circulation generated the observed behaviours of living things. In the 1650s for instance the French philosopher Descartes was convinced that all animals were dumb clockwork machines. Humans, special in the eyes of his Christian god, were granted an immortal soul that gave them superiority over mere beasts and the faculty of reason.

Cells. Today there are still workable definitions of life made by listing its properties. These aren't very different from Aristotle's, although usually souls are now set aside. But some major scientific discoveries have facilitated new approaches. For instance, his early microscopes allowed 17th C. Englishman, Robert Hooke, to discover the presence of *cells*, tiny room-like spaces, in cork. Dutch microscopist Antoni van Leeuwenhoek discovered microorganisms that were, in their entirety, single cells. The Dutchman was the first person to observe bacteria. Discoveries like these paved the way for the "cellular definition of life" – anything built of cells is an organism. But what then of a "dead" cell? How did it differ from one that was, by its movement for instance, apparently living? Even a deceased cat is built of cells.

Evolution. Since Charles Darwin and Alfred Russel Wallace published their theories of evolution in the 19th century, it has been suggested we might define life by reference to heritability and lineage: an organism is an entity with the potential to form a link in an evolutionary chain. That is, it was born of parents from which it inherited properties, and it has the potential to give birth to offspring to continue the inheritance chain. This rules out creatures born sterile – there still seems to be life in a mule, despite its infertility and legendary unwillingness to move. Like many definitions, the evolutionary approach is useful as a guide, but exceptions exist.

Genes and information. During the 19th century, the concept of the gene was introduced by Gregor Mendel, famous for breeding pea plants and documenting the inheritance of traits across generations. Early in the 20th century came the discovery by Sutton and Boveri that chromosomes were the means by which this inheritance was conducted. During the first half

of the 20th century it gradually became clear that DNA was the molecule of heredity. Its structure, the famous double-helix, was first published in 1953 by Francis Crick and James Watson. As part of their evidence they had used an X-ray diffraction image taken by Rosalind Franklin and Raymond Gosling the year before. This important image was used without the photographers' permission or knowledge in what has come to be one of the most famous scandals of modern science.

The decades of detective work into the mechanisms of inheritance permitted a new definition of life: an organism is a self-reproducing machine that stores its blueprint in a DNA molecule. Unfortunately this suffers from problems like those we have already encountered. Still, for a computer scientist this definition is particularly interesting as it opens the way to understanding life in terms of information storage and utilisation within an organism, and its transfer along evolutionary lineages. "Information", in a technical sense, is very much a concern of theoretical and practical computer science. In the 1940s and 50s, when computer science was taking its first steps under people like Alan Turing and John von Neumann, the possibility of viewing organisms as biological information processing machines, and of realising organisms through computation, was also taking shape. These ideas are still relevant in assessing the complexity of organisms and trying to understand the sense in which evolution might be driving increases in organismic complexity.

I have saved what I believe to be the most satisfactory approach to defining *life* until last.

Autopoiesis. An organism is a self-constructing, self-organising, self-bounding chemical machine defined as a set of processes of transformation and destruction of components. It exchanges material and energy with its environment (i.e. it is an open system) to maintain its internal relations and its existence as a physical machine. This explanation is roughly based on that offered in 1972 by Chilean biologists Humberto Maturana and Francisco Varela. They coined the term *autopoiesis* to capture their definition in a word – an organism is *self (auto) - made (poiesis)*.

The special thing about Maturana and Varela's idea is that the organism is defined as a set of relationships between its *processes*, not its static components. These processes involve the chemical transformation and destruction of molecules, and they are responsible for maintaining themselves, and for giving physical form to the organism from *within*.

Can an entity that meets Maturana and Varela's definition be realised in software? No, as it would not be a chemical machine, whatever other properties it had. But the possibility of building non-chemical autopoietic machines in digital media remains interesting to many and has the potential to inform us about the behaviour of life. Several researchers have attempted this by building "Artificial Chemistries" designed to support virtual autopoiesis. The catch is that the complexity of any real organism's network of processes is mind-boggling.

Only very simple metabolisms can currently be simulated in their entirety.

As noted earlier, there is *so* much that could be said about the history of attempts to answer the question, *What is life?,* that it could fill a library. Instead of writing one, since the concern of this book is specifically to explore Artificial Life's concrete forms, we will next describe some historical examples of the machines built to poke, prod, stretch, tear and crush theories of life. Argument about definitions is challenging and helps to clarify our thoughts, but if we could meet the challenge to build a machine that lived, that too would say something about our concept *life*.

Further reading

Maturana, H. R. and F. J. Varela (1972). "Autopoiesis and Cognition: The Realization of the Living". Dordrecht, Holland, D. Reidel Publishing Co.

McMullin, B. and F. J. Varela (1997). Rediscovering Computational Autopoiesis. Fourth European Conference on Artificial Life, MIT Press: 38-47.

Morange, M. (2008). "Life explained". New Haven / Paris, Yale University Press / Éditions Odile Jacob.

Russell, S. J. and P. Norvig (2010). "Artificial Intelligence, A Modern Approach", Prentice Hall, Chapter 1.

Schrödinger, E. (1944). "What is Life? The physical aspect of the living cell". Cambridge, United Kingdom, Cambridge University Press.

SECTION 2

Stone, bone, clay and pigment

KEY ARCHAEOLOGICAL FINDS

1. The Makapansgat cobble (South Africa)
   2.8M years BP
2. The Venus of Tan Tan (Morocco)
   300-500k years BP
3. *The Venus of Berekshat Ram* (Golan Heights)
   250-280 years BP
4. Chauvet cave paintings (France)
   30k years BP
5. Lascaux cave paintings (France)
   15k years BP

There is little point in speculating how, or even if, our earliest hominid ancestors thought about distinctions between living things and non-living things. We will probably never know. However we do know, through archaeological evidence, that entities we *now* consider to be among a class we label "living" were singled out for special attention even by our very earliest ancestors. The technology they employed was aimed specifically at mimicking life's appearance, i.e., the earliest artificial life forms appeal primarily to our visual sense.

The *Makapansgat cobble* is a face-shaped jasperite stone. This is currently the oldest lifelike (but essentially inanimate) object we have uncovered for which there is proof of its interest to early hominids. The cobble appears to have been collected by *Australopithecus* or a similarly ancient hominid, about 2.8 million years ago, presumably for its shape. The stone's resemblance to a hominid face is however, a natural fluke, it does not appear to have been altered manually. The Makapansgat cobble isn't therefore an early attempt to represent living things, but it is some evidence that its collector recognised the features of a hominid face in an unusual context, and found them interesting enough to collect.

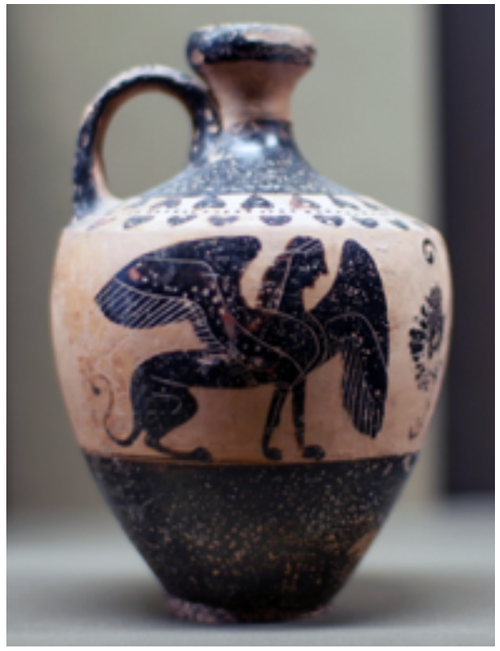
Two very early "venus" figurines, small stones of roughly hominid form, have been found that count as genuine artefacts; they have been crudely modified by hominids to accentuate their shape. These are the *Venus of Tan Tan* (Morocco, 300-500k years BP) and the *Venus of Berekshat Ram* (the Golan Heights, 250-280k years BP). Although the pebbles originally resembled hominid figures, if the theories that these were

modified are correct, then it seems reasonable to label the artefacts as the very first instances of artificial life. Their rough morphologies are a direct result of the application of simple, very simple, technology.

Leading into the Upper Palaeolithic period the artistic practice of our ancestors across several continents was unmistakably sophisticated. Many exquisitely fashioned ivory and bone figurines representing animals and humans have been recovered, especially from Europe's Jura mountains. Also spectacular, the caves of Chauvet and Lascaux in southern France were decorated around 30k and 15k years BP respectively with countless wall paintings. Between them the murals in these passages and caverns include horses, bison, lions, bears, panthers and many more lovely beasts.

Detail from the paintings at Lascaux caves reproduced on a wall of the *Gallery of Paleontology and Comparative Anatomy,* Paris.

At Chauvet there is, for the period, an unusual drawing dubbed *The Venus and The Sorcerer*. This appears to show a female pubic triangle and legs over which a minotaur-like figure looms – perhaps the first of many millennia of later fusions of animals and beasts – *chimaera*. This trope has maintained its cultural significance in mythology and art to the present day. The first fantastical animal/human fusion is truly an important step for artificial life as it is early recognition of the possibility for living things outside of direct experience. In fact, even in 1987 when the first academic conference officially called "Artificial Life" was organised by American researcher Chris Langton, he suggested the field be an exploration of "life as it could be". In their own way, this is something our ancestors considered in the depths of Chauvet cave.



A sphinx, Athens (560-550 CE) Musée du Louvre E718.

Certainly the aims and beliefs of the people of the Upper Palaeolithic would have been unlike our own. But considered coarsely, they too must have recognised a difference between themselves, animals and the inanimate. They may have situated the boundaries differently to us, for instance rivers, waves, clouds, planets and stars might have been lumped together as "self-movers", a term later popular with classical Greek thinkers. Regardless, collectively the representational finds we have are evidence of the complexity with which visual

representations in two and three dimensions were made of animals and hominids. Such work clearly drew many hours of skilled labour across the globe. It did this for many thousands of years before the advent of civilisation in Mesopotamia, Egypt or Greece. It is therefore no stretch to say that even with the simplest technology imaginable, life's representation has been at the forefront of conscious minds. The manufacture of visual likenesses using painting and sculpture continues unabated to the present day.

Further reading

Oakley, K.P., Emergence of Higher Thought 3.0-0.2 Ma B.P. *Philosophical Transactions of the Royal Society of London*. Series B, Biological Sciences, 1981. 292(1057): p. 205-211.

Bednarik, R.G., A figurine from the African Acheulian. *Current Anthroplogy*, 2003. 44(3): p. 405-413.

d'Errico, F. and A. Nowell, A New Look at the Berekhat Ram Figurine: Implications for the Origins of Symbolism. *Cambridge Archaeological Journal*, 2000. 10(1): p. 123-167.

SECTION 3

Wheels and joints

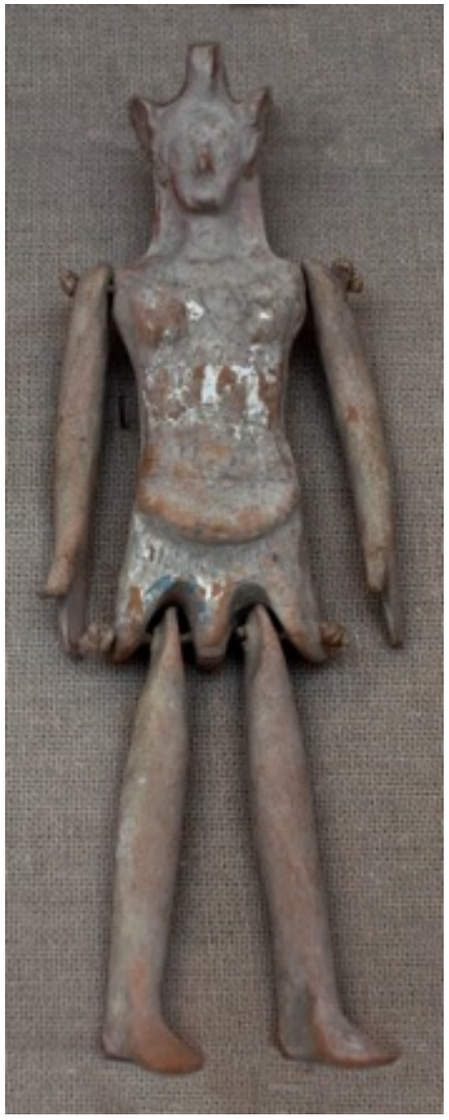
TOPICS

1. Articulated dolls
2. Wheeled animals
3. Puppets and marionettes

Clay-bodied toy mouse with moving tail and mouth of wood. Egypt, New Kingdom (1550-1070 BCE). British Museum EA 65512

A toy cat (or lion) with an articulated jaw operated by tugging a string. From Thebes, Egypt. New Kingdom (1550-1070 BCE). British Museum EA 15671

A visit to a well-stocked toy shop today reveals countless shelves of brightly coloured playthings, many of which have equivalents dating back to ancient Mesopotamia, Egypt and Greece. Now as then, many of the toys replicate the appearance of animals, a few extend this to replicate some aspect of animal motion. Examples include a very cute hedgehog, lion and a bird on little wooden carts dating from between 1500 and 1200 BCE. The Louvre displays a Greek buffalo from antiquity that does away with the cart altogether. Its axles are mounted directly through its body where its legs ought to be. An Egyptian cat and a Nile crocodile, both with articulated jaws, have also been recovered. This same mechanical snap, coupled with the mobility provided by the wheels of the buffalo, amuses toddlers 4000 years later as they trundle their toys around the floors of the world.



Articulated terracotta *Pyrrhic dancer*; doll dressed in helmet and cuirass. Greek, Corinth (c. 450 BCE). British Museum GR 1930.12-17.5. BM Cat Terracottas 930.

Another current toy with ancient origins is the articulated doll. These have been produced with jointed legs, knees and

Buffalo on wheels. Ancient Greece (probably 800–480 BCE). Louvre, Collection Campana, 1861 Cp 4699.

sometimes shoulders at least since 700 BCE. Like jointed dolls, the origins of marionettes and other puppets stretch back to antiquity, especially in Asia and Greece where they also found favour as a basis for philosophising about human behaviour. For instance there is a tale from China of a shadow puppet being used to console a king whose favourite concubine was recently deceased. In Greek philosophy Plato refers to living creatures as sinew and cord driven puppets of the gods, Aristotle uses the mechanisms of self-moving puppets as an analogy to assist him in his explanation of an animal's response to stimuli.

Throughout history, replicas of animal motion have continued to fascinate us. The current attempts to mimic it extend into the virtual world of computer animation and graphics, but life-sized animatronics remain popular in certain contexts – for instance, as tacky showpieces to frighten museum visitors with the gaze and roar of Tyrannosaurus Rex. Roboticists too are interested in animal motion as inspiration for their own **autonomous** mobile machinery. This has applications in exploration, rescue, entertainment and (sadly) warfare since animals are often well adapted for motion through complex natural terrain.

Terracotta horses and cart. Daunian, South Italy (750-600 BCE). British Museum GR 1865 1-3.53

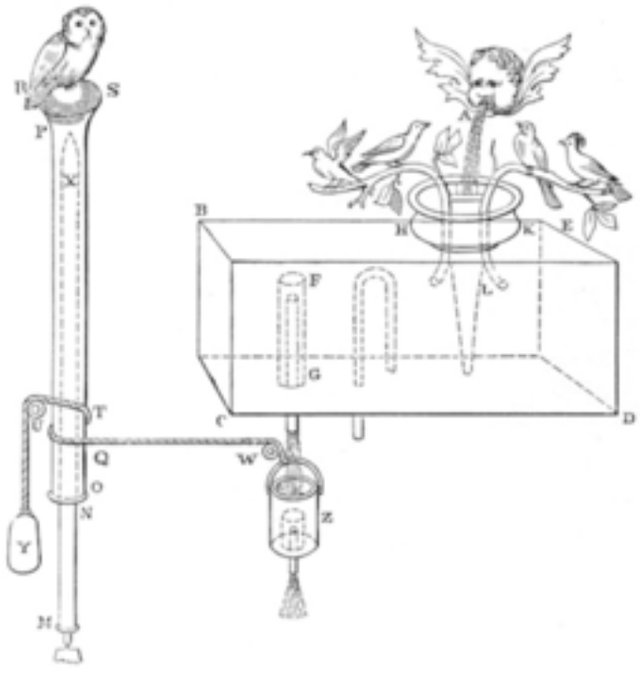
A toy wooden horse that originally ran on wheels/axles mounted through its feet. Roman period, Oxyrhynchus, Egypt. British Museum GR 1906.10-22.14

SECTION 4

Breath

PNEUMATIC ENGINEERS OF ANCIENT GREECE

1. Ctesibius of Alexandria (fl. c. 270 BCE)
2. Philo of Byzantium (c. 280 - 220 BCE)
3. Hero of Alexandria (c. 10 - 70 CE)



"Birds made to sing and be silent alternately by flowing water", Pneumatics (1st C CE), in "The Pneumatics of Hero of Alexandria" (1851), Taylor Walton and Maberly publishers, trans. Woodcroft. Image out of copyright.

Many cultures consider their creation deity to have built humans of earth, clay, stone or mud, sometimes this was thought to have been performed quite literally on a potter's wheel or in a potter's workshop. Examples of deities working with earthly media arise in the stories of ancient Egypt, China, the Middle East, Africa, Greece and Rome to name a few. This is hardly surprising given, as we have seen, that the materials had been known for around 3 million years to be capable of holding human form. But what distinguishes inert earth from animated creatures? Why didn't statues just get up and move about? The missing ingredient was breath! According to several cultures this was placed by the deity in the inert sculpted earth and only departed at death.

Come hither, all you drinkers of pure wine,-
Come, and within this shrine behold this rhytum,
The cup of fair Arsinoe Zephyritis,
The true Egyptian Besas, which pours forth
Shrill sounds, whenever its stream is opened wide,-
No sound of war; but from its golden mouth
It gives a signal for delight and feasting,
Such as the Nile, the king of flowing rivers,
Pours as its melody from its holy shrines,
Dear to the priests of sacred mysteries.
But honour this invention of Ctesibius,
And come, O youths, to fair Arsinoe's temple.

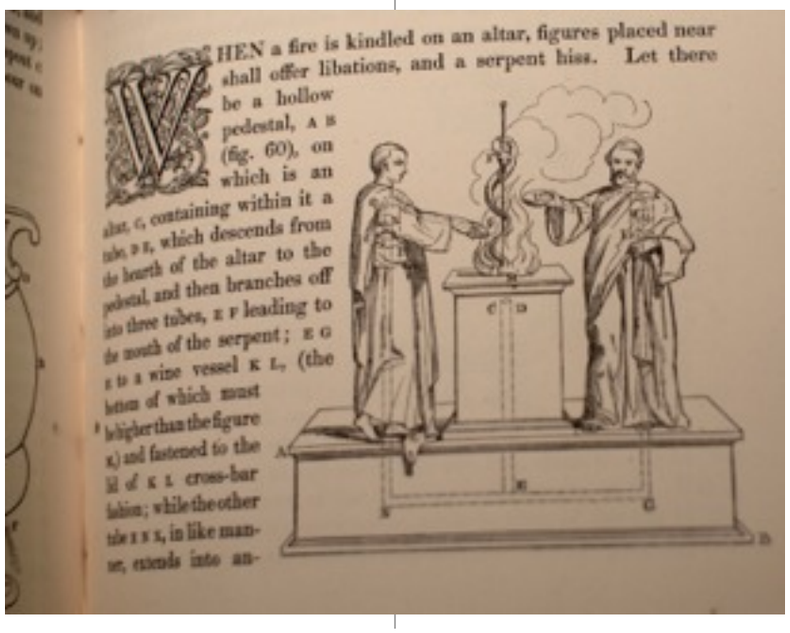
Athenaeus, 3rd C. CE

Perhaps, thought some inventors of ancient Greece, breath might be made to emanate from our own technological creations? This idea proved a viable means of demonstrating the

principles of a field that was known in ancient Greece as *Pneumatics*; the study of pressure, especially as it related to the movement of air and water. An added complication was the debate concerning a "philosophical" *pneuma*. This mysterious substance was at least partially responsible, according to the thinking of the time, for animating living bodies. It was quite distinct from air, but the linguistic doubling betrays a similarity between the two materials that lends the pneumatic devices an air of lifelikeness. Pneuma, even in living human tissue, was often very closely associated with physical air.

Ctesibius

The inventor Ctesibius (fl. c. 270 BCE), alluded to above in Athenaeus' verse, was an early pneumatic experimentalist from ancient Greece who was very interested in making devices that could move themselves – *automata*, literally, self-movers. The breath, a whisper, a voice, a kiss; these are all associated with life and it was therefore no surprise that his experiments often took the form of statues that appeared to live. A breakthrough in his work, in fact in technology generally, was the manufacture of water-clocks (called at the time, *clepsydrae*, literally, "water thieves") with a regulated flow. He used his improved clepsydrae to power simple moving figures and to sound hourly alarms. Sometimes Ctesibius' clock figures blew trumpets or whistles, a trick managed by rapidly filling a vessel with water, causing air to be evacuated through the appropriate sounding mechanism. Other devices had small figures that rose and fell along graduated columns holding pointers indicating the time.


HEN a fire is kindled on an altar, figures placed near
shall offer libations, and a serpent hiss. Let there
be a hollow
pedestal, A B
(fig. 60), on
which is an
altar, C, containing within it a
tube, D E, which descends from
the hearth of the altar to the
pedestal, and then branches off
into three tubes, E F leading to
the mouth of the serpent; E G
E to a wine vessel K L, (the
bottom of which must
be higher than the figure
K) and fastened to the
ld of K L cross-bar
fashion; while the other
tube E H E, in like man-
ner, extends into an-

"Libations poured on an Altar, and a Serpent made to hiss, by the Action of Fire." Sect. 60 of Pneumatics (1st C CE) in "The Pneumatics of Hero of Alexandria" (1851), Taylor Walton and Maberly publishers, trans. Woodcroft. Image out of copyright.

Philo

Philo of Byzantium (c. 280 - 220 BCE) continued working in the tradition of Ctesibius. He invented a mechanical maid that dispensed wine when a cup was placed in her hand, then mixed the wine with water to make it suitable for drinking.

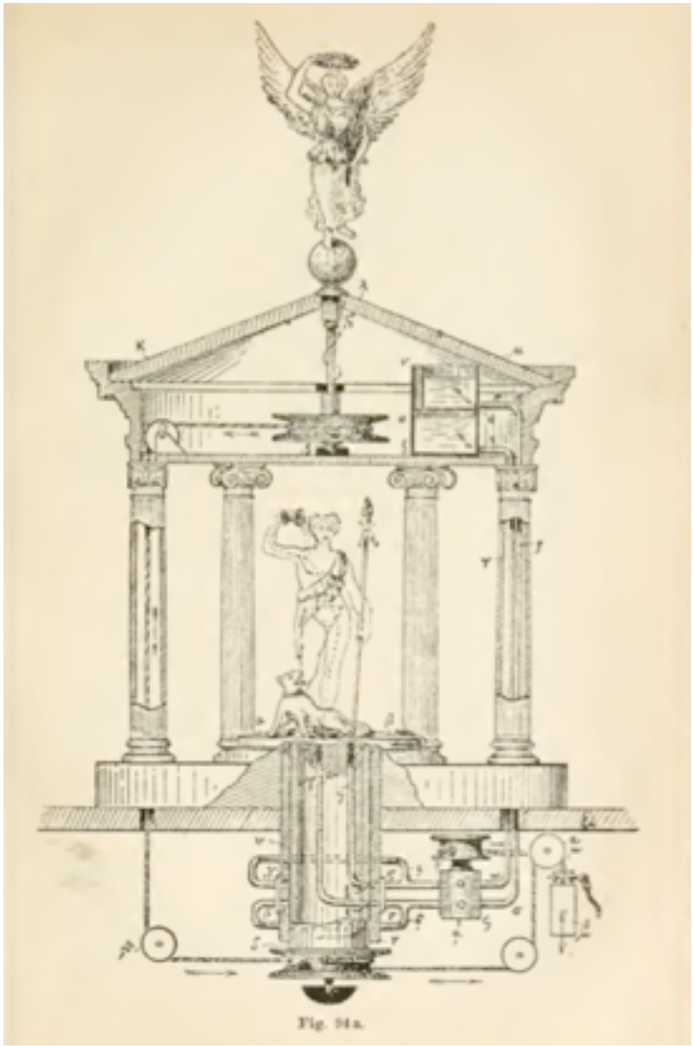
Philo created another vessel that supported a metal tree upon which a cleverly wrought mother bird nested, shielding her chicks from harm. As water or wine was poured into the vessel, a snake, driven out by the liquid, alarmingly rose towards the mother's brood. When it approached too closely, the mother ascended above her family, spread her wings and frightened the snake back into its hole. She then folded her wings and returned to her young. The whole device was driven by concealed floats suspended in the liquid. Using this simple technology Philo had imitated the behavioural response of a protective animal in danger.

Hero

Perhaps the most famous ancient Greek pneumatics practitioner was Hero of Alexandria (c. 10-70 CE). Like Philo, he devised many animated creatures driven by falling water and pressurised air. These included hissing serpents and dragons and trick vessels with animals that slurped liquids from them, or deposited liquids into them when the water level fell. Some of Hero's automated dioramas replicated the action and response seen in the work of his predecessor. For instance, Hero devised a water basin surrounded by twittering metal birds. Nearby, sat an owl with its back turned. At regular intervals the owl would cast its gaze towards the twittering birds, who would fall silent, until once more the owl turned its back on them and they could resume their twittering unthreatened.

Hero also devised two automatic miniature theaters. Each consisted of a solid plinth upon which were positioned little animated figures that mechanically enacted a series of scenes. The theaters were powered internally by a falling weight.

The fall of the weight was slowed and regulated by sitting it within a grain filled clepsydra that operated like an hourglass. One of these theatres could move onto a large performance stage by itself – rolling from offstage then coming to a halt in the limelight. It would give an animated performance that included altar fires being lit and extinguished, a dance to Bacchus (the god of wine) with flowers and flowing alcohol. Automatically generated sound effects enhanced the show. The contraption would then mysteriously roll back off stage of its own accord.



The stage of Hero's automatic, self-moving theatre from Schmidt, W. (1899). "Herons von Alexandria, Druckwerke und Automatentheater". Leipzig, Druck und Verlag von B.G. Teubner. Image out of copyright.

Remarkably, Hero designed a programmable self-moving cart. This ingenious automaton was not only capable of forward and reverse programmed movements, if set up properly it could execute complex turning motions typical of animals. The steering mechanism was controlled by a dual cord tugged by a concealed falling weight in the same manner as his theaters. The two cords were wrapped individually around the two halves of the cart's split rear axle, enabling independent drive for each wheel. Each wheel's axle could be studded at different positions around its circumference with wooden pegs. By winding the cord around a peg, Hero could alter the direction an axle

was spun as the cord was tugged by the falling weight. This reversal allowed him to change the direction each wheel rotated independently of the other, enabling the vehicle to be “programmed” to turn as desired. Pauses could be added to the motion by leaving slack in the cord between axle-pegs.

Summary

The pneumatic inventions of the ancient Greeks were probably not intended to *be* living things, as much as they seemed to mimic animal and human behaviour. Many of them were presented as exciting demonstrations of the principles of pneumatics by engineers with a flair for the theatrical. Some of the devices may have been used in temples to attract and impress visitors. Others could have appeared on dinner tables for conversation starters at the drinking parties of the well-to-do. The inventions were sometimes celebrated in verse and prose, for instance in the poem quoted earlier, but by and large they seem to have been forgotten as the centuries saw the decline of ancient Greece. Eventually, a few Arab inventors came across the manuscripts of the Greeks and began to explore the wonderful descriptions and instructions they found – adding a unique and exciting Islamic flourish.

Further reading

Vitruvius (c. 15 BC). The Ten Books on Architecture. Cambridge, Harvard University Press, 1914. (9:8:2-7, 10:7-8 for information on Ctesibius.)

Prager, F. D. (1974). "Philo of Byzantium, Pneumatica, The first treatise on experimental physics: western version and eastern version", Dr. Ludwig Reichert Verlag Wiesbaden.

Hero of Alexandria (1st C CE). "The Pneumatics of Hero of Alexandria". London, Taylor Walton and Maberly, trans. Bennet Woodcroft, 1851.

Schmidt, W. (1899). "Herons von Alexandria, Druckwerke und Automatentheater". Leipzig, Druck und Verlag von B.G. Teubner. (In Greek and German.)

SECTION 5

Islamic and Chinese automata

AUTOMATON ENGINEERS OF THE MEDIEVAL ISLAMIC AND CHINESE WORLDS

1. Banū Mūsa bin Shākir – the brothers Mūsa (c. 850 CE)
2. Ibn Khalaf al-Murādī (1266, orig. 11th C. CE)
3. Badi'al-Zaman Abū al-'Izz Ismā'īl ibn al-Razāz al-Jazarī (1136 - 1206 CE)
4. Su Sung (1020 - 1101 CE)

Perhaps the most famous of the Arab automaton engineers was Badi'al-Zaman Abū al-'Izz Ismā'īl ibn al-Razāz al-Jazarī, or just al-Jazari for short (1136 to 1206 CE). He was employed by the rulers of a northern province of the Jazira region in Upper Mesopotamia to preserve his knowledge of automata in *The Book of Knowledge of Ingenious Mechanical Devices* (1206).

In the Arab world an earlier text, *The Book of Ingenious Devices* had been produced in Baghdad by the Banū Mūsa bin Shākir (c. 850 CE). This had demonstrated many of the pneumatic principles described by Hero of Alexandria, primarily using various kinds of trick wine and water vessels for drinking applications. The devices in this earlier Arab text were not much concerned with the animation of figures. Al-Jazari however, was keenly interested in automata. He added to this a practical engineering focus that had not been seen even in the work of the ancient Greeks. Al-Jazari took pride in the fact that he had personally tested the workings of his machines. They were supposed to be functional ingenious devices for hand washing, wine serving, party entertainment and telling the time. Al-Jazari even describes a few for the (hopefully reliable!) measurement of blood let from a patient's arm. These are particularly interesting for the way they employ simple mechanical human scribes to count units, and tens of units, of blood.

Among al-Jazari's devices are many intricate human and animal automata. In a unique un-pneumatic vein, al-Jazari describes the manufacture of a calibrated candle-clock based

Al-Jazari's candle clock with wick-trimming swordsman and ball-dropping falcon. Image from MS held by the Smithsonian Inst. Early 14th century (1315 CE, December) /Dated Ramadan 715. Farruk ibn Abd al-Latif (CB). Opaque watercolor, ink, and gold on paper (H: 30.8 W: 21.9 cm), Syria. Purchase, F1930.71. Image out of copyright.

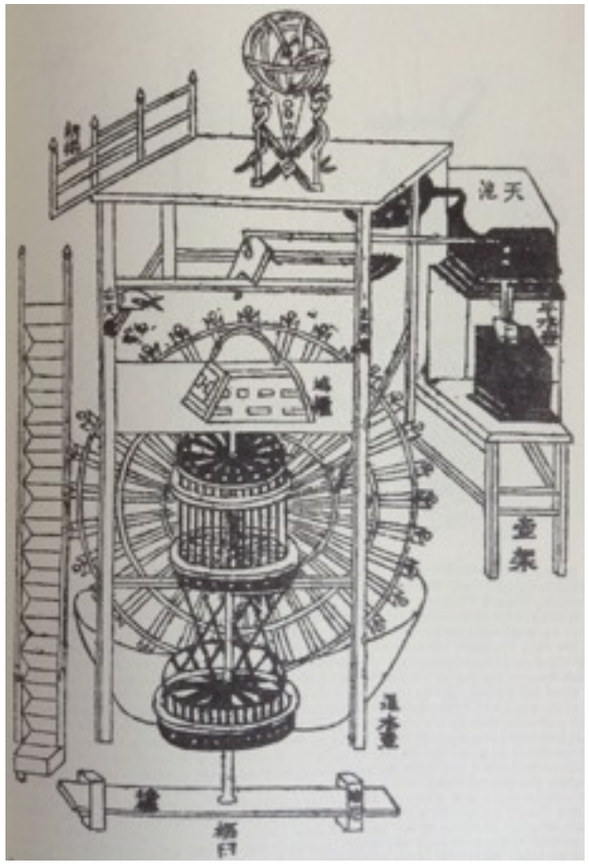
around a tall brass candle-holder. This has a falcon, its wings outstretched, perched at its base. Towards the top of the candle-holder a shelf supports a sword-bearing slave. The candle is topped with a metal cap for the wick to pass through. As the candle is consumed by the flame, it is automatically raised to ensure that the lit wick always protrudes through the cap. During the night the mechanical swordsman slashes at the wick, trimming any excess. At the swordsman's stroke, a metal ball falls from the mouth of the falcon into a container at its feet. By counting the balls it is possible to tell the hour.

Like his predecessors in Greece, al-Jazari describes several water-clocks interpreted with the assistance of animated figures. Although he is probably best know for his enormous and very animated Elephant Clock (an 8.5m replica has been installed in the Ibn Battuta shopping mall in Dubai), I prefer another.

My favourite of al-Jazari's water clocks involves a peacock, peahen and two chicks driven by a constant flow clepsydra. The peahen rotates in an arc, her beak smoothly marks the passing of half an hour. A series of 15 roundels forming an arch above the scene change colour from white to red to indicate the passing of half hours. The peace is interrupted every half hour as the two chicks whistle and peck at one another. The peacock turns slowly with his tail upright in display. Finally, the peahen's orientation is automatically reset so that the arc of the next half hour can be swept out by her beak like the last.

Another exciting Arab text that includes automata is *The Book of Secrets in the Results of Ideas* (1266 from an 11th C. original) by Ibn Khalaf al-Murādī. This was probably written in Cordoba in the south of Spain. Among other machines, it describes intricate designs for clocks that schedule multiple human and animal figures and also miniature animated vignettes that sequence characters to tell scenes from fairytales.

Al-Muradi's figures typically emerge from miniature doors before moving about in various ways. The doors conveniently hide the figures from view between performances. This is a feature that appears on many early Islamic clock designs and is reminiscent of the shutter on much more recent cuckoo clocks originating in south western Germany. Once outside their housing, the automata of al-Muradi may enact a short scene



The water-powered astronomical clock tower of Su Sung, c. 1090 CE. (Hsin Fa Yao, ch. 3, p. 4a). Reproduced from Joseph Needham, *Science and Civilization in China*, Vol. 4, Pt. 2, Mechanical Engineering, Cambridge Uni. Press, 1965, p. 451. Image out of copyright.

such as a sword fight, chase, or perhaps simply an action and response such as the appearance of a threat and the retreat of a fearful character. In one case, a surprise villain attempts to assassinate two women who rapidly, and safely, retreat to their housing. In a longer mechanical episode, a boy appears from his hiding place in a well to call for a lovely girl. Four gazelles arrive to drink as the object of the boy's affections emerges from the doors of her home. At this, three venomous snakes set by the boy's rival appear and frighten the gazelles away. The boy himself retreats into the well where he hides until the snakes depart.

Coloured roundels and the dropped metal balls employed by al-Jazari to allow a viewer to read the time are found likewise in several of al-Muradi's works, including some that stage animated plays. Hence, as well as providing miniature theatre, some of the devices had practical applications.

During the period from the Banū Mūsa to al-Jazari, the Chinese were also constructing complex mechanical devices adorned with automata. Perhaps the best known is the grand Chinese astronomical clock of Su Sung. This multi-storied tower, built c. 1090 CE, sported literally dozens of animated figures to sound the time, *jacks* as they later became known in Europe. The clock also included a large armillary sphere and a celestial globe for the serious business of tracking the heavens.

Further reading

Ahmad Ibn Khalaf al-Murādī, (1266). "The Book of Secrets in the Results of Ideas". Trans. A. Ragab, Milan, Italy, Leonardo3.

Badi'al-Zaman Abu'l-Izz Isma'il b. al-Razzaz al-Jazari, (1206). "The Book of Knowledge of Ingenious Mechanical Devices". Trans. D.R. Hill, Dordrecht Holland, Boston U.S.A, D. Reidel.

Banū Mūsa bin Shākir, (c. 850 CE). "The Book of Ingenious Devices". Trans. D.R. Hill, Dordrecht, Holland, D. Reidel.

Needham, J., W. Ling and D. J. d. S. Price (1960). "Heavenly Clockwork, the great astronomical clocks of medieval China". Cambridge, UK, Cambridge University Press.

SECTION 6

Clockwork

TOPICS

1. The origins of the clockwork metaphor
2. Clock-mounted automata
3. Automata gain their independence

This dulcimer player was built in the late 18th century by David Roentgen and Pierre Kintzing, possibly as a representation of Marie Antoinette. It is one of many androids (human-like automata) in the collection of the Conservatoire National des Arts et Métiers, Paris.

The origins of the clockwork metaphor

As we have seen, the clocks of ancient Egypt, Greece, Rome, the Islamic empire and China usually required a continuous stream of liquid to be provided. This was measured against a scale to tell the time, or its weight was used to trigger a discrete event such as the tipping of a bucket or the turning of a wheel. This event might in turn trigger the clanging capture of a steel ball in a tin pan, or the rotation of a gear in a more complex machine, to mark time audibly or by the movement of a figure.

This is an early image of a verge escapement from Giovanni de Dondi's astronomical clock, the *Astrarium*, 1364, Padua, Italy. The image appeared in de Dondi's *Tractatus Astrarii*.

Late in the 13th century, a *verge escapement* was invented. This mechanical device repeatedly and regularly halts and releases a toothed wheel that might be driven to rotate by a falling weight or coiled spring. Instead of allowing the toothed wheel to spin rapidly, exhausting the energy stored in the system, the regular halting and releasing turns what would otherwise be a rapid continuous unravelling, into the oscillation of a pendulum or a sprung **balance wheel**. The oscillatory motion can be used to drive the hands of a clock or the parts of an automaton. With improvements in mechanical escapement design came the possibility of manufacturing

highly accurate timepieces. The basis of the mechanism is still used in today's boutique wristwatches and ornamental clocks.

Over the five hundred years since the invention of the mechanical escapement, clockwork automata mimicked the movements of humans and other animals throughout Europe. In large part, their development was accelerated during the Renaissance by the rediscovery and translation of Hero of Alexandria's texts. These came to be known via earlier translations obtained through the Islamic empire.

A *deadbeat* escapement mechanism. The toothed wheel is driven to rotate via a gear train that is activated by energy stored in a coiled spring or a suspended weight. The wheel's motion is repeatedly halted and released as the pallets interfere with it. This interference converts the checked spinning motion of the wheel into the regular oscillation of a pendulum that runs vertically off the bottom of the image.

In literature, science and philosophy, the clockwork mechanism became synonymous with the concept of *life*. The French philosopher René Descartes (1596-1650) quite literally believed animals to be god's clockwork automata. To Descartes' mind however, humans had an additional immortal, god-given soul that provided us with our faculty of reason and elevated us above the other creations. To Julian Offray de La Mettrie (1709–1751), another French philosopher (and physician), it seemed possible that the *whole* of human behaviour might be explained by reference to mechanistic processes. Soon after La Mettrie's time, perhaps arriving with the German Romantics of the early 19th century, the concept of the automaton came to imply a meticulous, time-obsessed, uncreative lack of spontaneity. To be an "automaton" was, in that time as now, to be sub-human. This didn't stop the development of incredible works of mechanical artificial life, the predecessors of today's robots. Many of these came from European clockmakers and inventors. Some arose in Japan which was later to become a centre for robotics culture, research and development.

Clock-mounted automata

Some of the earliest automata operated by completely mechanical escapements and drive mechanisms appeared on European town clocks. These were situated in public squares as important civil regulators and displays of city wealth. A common form of clock automaton was the "jack" or *jacquemart*, a mechanical bell-ringer that chimed the hours.

Revolving Jacks that survive from 16th century Netherlands, now in the collection of the British Museum, London.

The *Orloj*, an astronomical clock installed in Prague's old town square in the early 15th century, still presents today an entire parade of automata. Two windows open above its large blue dial, an astronomical clock face, and a procession of twelve apostles begins. Each turns to face the crowd through a little portal before making way for

Jacks strike the bell on this Medieval clock from the Netherlands, 1450-1550. The clock is now in the collection of the Museum Speelklok, Utrecht, Netherlands.

the next. The figure of Death noisily tolls his bell, an hourglass sits in his other hand as a reminder that the sands never cease to run. Beside Death, a Turk wearing a turban strums his lute and nods his head in time to the performance. Across the clock, Vanity admires his appearance in a hand-mirror and a Jew simultaneously taps his cane, shakes his head and jingles a bag of gold. With its Turk and Jew the Orloj provides a daily reminder that our race relations have not progressed far since the Middle Ages, but also that automata can, and have, played significant sociopolitical and cultural roles.

Detail of clock modelled after the Strasbourg cathedral clock (1589). Top layer down: Death strikes the hours beside Jesus; the four ages of Man strike quarter hours; angels proceed before a seated Madonna and child while music plays; days of the week are represented by a parade of the planets. Collection of the British Museum, London.

Automata gain independence

Many mechanical marvels were produced for appearances in cathedrals without reference to timekeeping. Much as the constructions of Hero of Alexandria might have been used in classical temples, for awhile, some European places of worship used automata as a means to attract people to their services. This didn't last long. Sadly, the "false idols" and unsavoury gimmicks were eventually destroyed by zealots. A wind-up automaton monk dating from the mid 16th century survives in the collection of the Smithsonian Institution. This figure walks about, striking his chest with an arm as he waves a wooden cross and rosary in the other. His eyes roll around in their sockets and his lips move silently as his head turns and nods. Sometimes the monk kisses his little cross as he prays for the owner, devoutly, automatically and *ad infinitum*.

The town clock in Bern, Switzerland is situated in the Zytglogge (time bell) medieval tower. This important civil building has housed a time-keeper since the early 15th century.

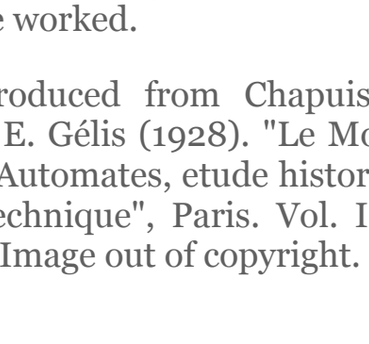
Automata were also made for display in the exclusive gardens and chambers of the aristocracy, and for public exhibitions. Even Leonardo da Vinci seems to have built a couple– a walking lion that produced a bouquet of lilies from a cavity in its chest, and a puppet-like mechanical knight. Some androids – automata of human appearance – could write, play musical instruments or draw pictures. Some were even made to utter sounds.

The Writing Hand pens an extract from the poetry of Virgil, “May God not impose ends or deadlines on this house”. It was invented by Friedrich von Knaus in 1764, dedicated to the House of Lorraine, rulers of Tuscany at the time. Now in Museo Galileo, Florence, Italy (Inv. #3195).

One of the most famous inventors of automata was Jacques de Vaucanson (1709–1782). Perhaps he is best known for his duck, an automaton that could eat a pellet offered to it, extend its neck as it swallowed, then defecate a foul smelling waste out the other end! This complex machine could flap its wings and wriggle about in what was, by reputable reports of the time, an astonishing marvel of clockwork invention. Vaucanson also created a life-sized flautist and a mechanical player of the pipe and drum. The figures did not fake their musical performances, they really played by breathing into and striking their instruments.

The last automata I will describe are *Karakuri Ningyo*. These are Japanese androids, mechanical robots designed to amuse and entertain. Particularly popular during the 19th century were tea-serving dolls, independently mobile clockwork figures that would enter a room and walk towards a guest, carrying with them a cup of tea. When the cup was lifted the automaton would stop. The guest might then drink the tea before returning the empty cup to the tray the automaton was carrying. At this, the automaton would turn and walk back out the way it had entered.

A master of Japanese automata, Hisashige Tanaka (1799-1880), produced a lovely archer boy. This figure sits placidly before his quiver. He looks downwards, draws an arrow, strings his bow, inspects the target... and fires. Several times. Honestly! This beautiful automaton still survives.



Vaucanson’s duck concealed a “fake” digestive system in the sense that it didn’t chemically breakdown its food or use it to build biomass. It could however, consume a pellet and wriggle about as if eating and swallowing it. A short time later a foul-smelling substance would be emitted from its rear end, much to its audience’s amazement and amusement! This image from 1738 is an attempt by one observer to suggest how it might have worked.

Reproduced from Chapuis, A. and E. Gélis (1928). "Le Monde des Automates, etude historique et technique", Paris. Vol. II, p. 151. Image out of copyright.

Further reading

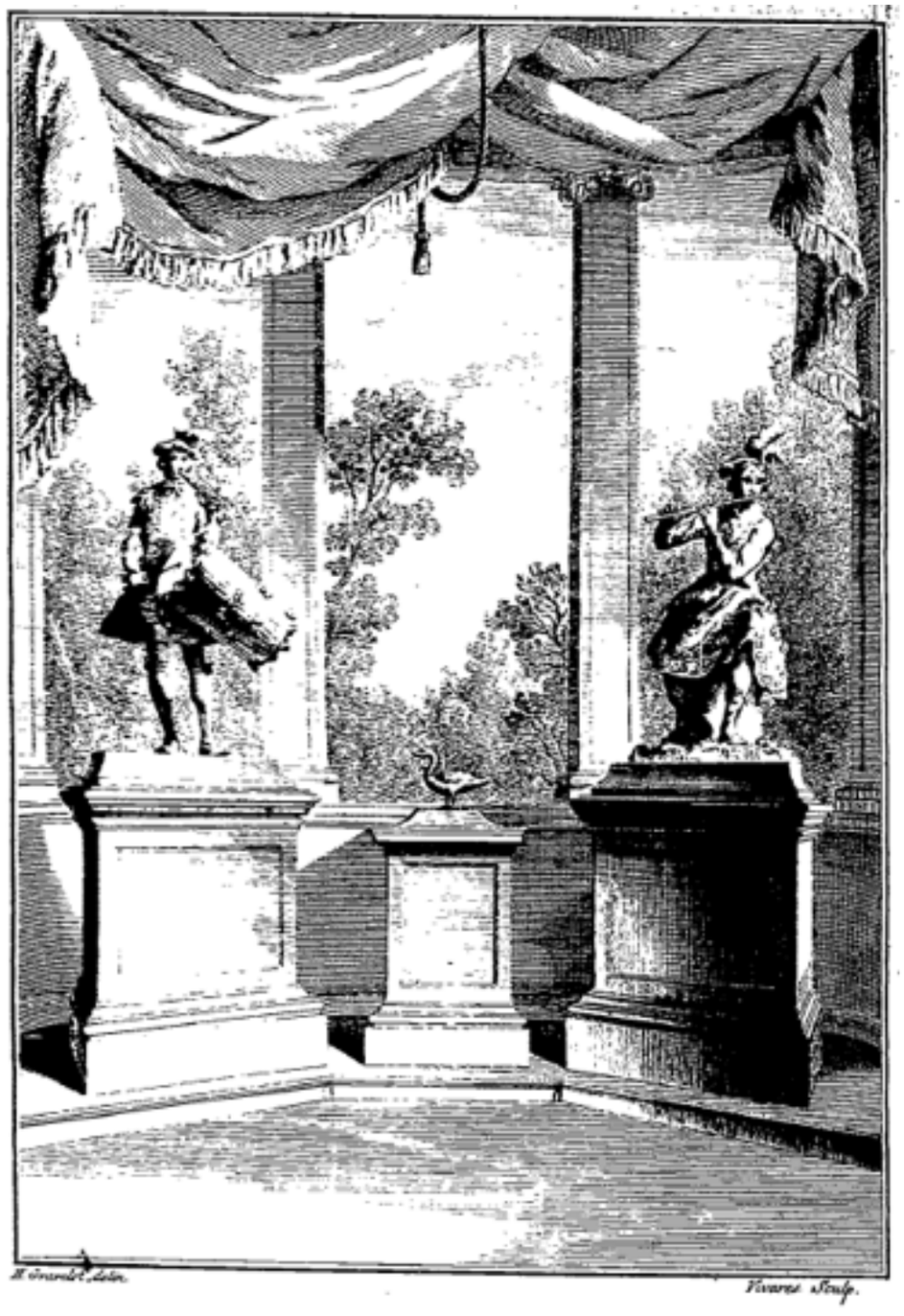
Kang, M. (2011). "Sublime Dreams of Living Machines: the automaton in the European imagination". Cambridge, Massachusetts; London, England, Harvard University Press.

Riskin, J., Ed. (2007). “Genesis Redux, Essays in the History and Philosophy of Artificial Life”. Chicago / London, University of Chicago Press.

Wood, G. (2003). "Living Dolls, A Magical History of the Quest for Mechanical Life". Kent, Faber and Faber.

Tippoo's Tiger simulates the groans and roars of the demise of a European. The work has European mechanical internals enclosed in a South Indian carving. It was created in the late 18th century. Collection of the V&A Museum, London.

The Settala devil. An automaton, possibly manufactured from a statue of Jesus, (16th or 17th C). The mechanism allows the head and eyes to turn, and for the devil to poke out its tongue and emit a strange sound. Collection of the Museum of Applied Arts of Castello Sforzesco, Milan, Italy.



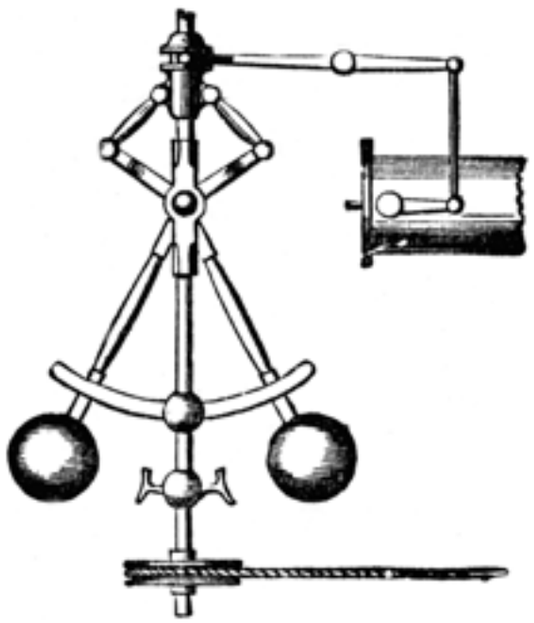
The frontspiece from an account of the mechanism of Jacques de Vaucanson's automata. Here are shown a life-sized flautist and a player on the pipe and drum. The account was presented to the Royal Academy of the Sciences, Paris, 1742.

SECTION 7

From steam to electricity

TOPICS

1. Heat and self-regulation
2. Cybernetics
3. Frogs legs and animal electricity
4. Robotics



This centrifugal governor is operated by a pair of "fly balls" that spin at a rate governed by the engine speed. As the engine speeds up the balls are flung outwards and upwards. This action operates control rods to throttle back the amount of steam entering the engine. As the influx of steam falls, the engine slows and the balls fall a little under gravity. Hence, the amount of steam entering the engine is increased slightly. This feedback/coupling between speed and fuel supply automatically regulates the engine.

Heat and self-regulation

The late 18th and early 19th century saw the beginning of the Industrial Revolution in Britain. Dependence on water as a power source fell as efficient steam engines of immense power began to operate factories day in, day out, and night after night. The period saw many transformations in mining, transportation, the textile and other manufacturing industries. With steam power came several new perspectives on machines and *life*. For instance organisms could be viewed as a form of "heat engine", machines transforming chemical energy into heat, and then into kinetic energy.

The engineers Matthew Boulton and James Watt, who were pivotal in the development of steam power, also developed *governors* to regulate the speed of steam engines. The governor provided important insights for understanding biological organisms as *dynamical systems*. Automatic governors operate on a "negative feedback loop" that reduces the rate at which fuel (steam entering the cylinders of a steam engine for instance) enters the engine as the speed of the engine increases. As the engine gets faster, the steam supply is reduced, as the engine gets slower, the steam supply is increased. The result of these frequent micro-adjustments is an engine running, without manual intervention, at a regulated speed. The engine is said to be a dynamical *homeostatic* system because it maintains one of its parameters, its speed in this case, constant. It does this by continuously adjusting its behaviour and in spite of external variations in its running conditions. Of course another way to maintain state in the face of external

perturbations is to be inert, like a rock. But we don't learn so much about organisms from stasis as we do by studying dynamics. This is clear in light of the concept of *autopoiesis* discussed earlier.

Organisms are homeostatic dynamical systems: mammals for instance, keep their body temperature approximately constant even as the environmental conditions fluctuate; they maintain constant salinity and hydration; the iris in the mammalian eye contracts and dilates to control the amount of light hitting the retina. All organisms maintain their organisation – the set of relations between their processes of self-production – constant. If they didn't do this they would cease to exist. Evolution too, the process by which new organism traits are introduced, modified or eliminated from biological populations, has been explicitly likened to a centrifugal governor by one of its central figures, Alfred Russel Wallace in his famous 1858 paper "On the tendency of varieties to depart indefinitely from the original type".

Cybernetics

Self-governance proved to be a valuable insight offering new ways of interpreting biology and ecology, and for the construction of artificial life. In particular, a field dubbed *Cybernetics*, arose in the 1950s, at what was to become the dawn of the Information Age. The field's practitioners, famously W. Ross Ashby, Norbert Wiener, Gordon Pask, and William Grey Walter, concerned themselves with communication and control systems in animals and machines. The cyberneticians were very interested in the possibilities of feedback loops and designed several iconic systems exhibiting homeostasis of one type or another. The precursors of the thinking that led to Artificial Life are also apparent in their writings:

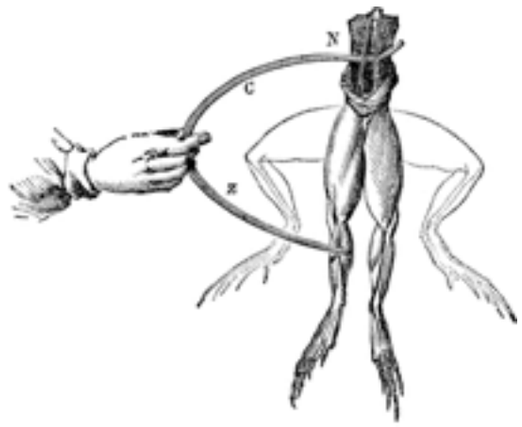
A cybernetic *Tortoise* designed by Grey-Walter (c. 1950) in the collection of the Science Museum, London. These mobile electric robots were equipped with lights, sensors and simple circuits that allowed them to exhibit many animal-like behaviours including following one another, fleeing, even returning to their hutch when they needed a recharge of their batteries. More recently, Valentino Braitenberg has designed a host of (conceptual) machines using similar principles. These are described in his book *Vehicles* (MIT Press, 1984).

> *Cybernetics stands to the real machine electronic, mechanical, neural, or economic — much as geometry stands to a real object in our terrestrial space... It takes as its subject-matter the domain of "all possible machines", and is only secondarily interested if informed that some of them have not yet been made, either by Man or by Nature. What cybernetics offers is the framework on which all individual machines may be ordered, related and understood* – Ashby, *An Introduction to Cybernetics,* 1956.

The artist Edward Ihnatowicz was one of many exhibiting works in the late 1960s and early 70s that were based on the

principles of Cybernetics. His *Senster* robot was a giant armature that responded to the presence of exhibition visitors in ways uncannily like those of a sentient being. His sculpture S.A.M. (Sound Activated Mobile) took on a less threatening but equally lifelike appearance. It resembled a potted flower that turned this way and that to engage humans who had come to talk to it on its plinth. A number of artworks in the landmark 1968 London exhibition curated by Jasia Reichardt, *Cybernetic Serendipity*, engaged their audiences in similar ways.

Alan Turing and John von Neumann, two fathers of modern day Computer Science, also contributed to the development of Cybernetics in different ways. Some of their work is detailed later in this book. In general, the field has played a significant role in establishing the position of the computer in Artificial Life and interactive technology. Unlike the modern digital computer, the majority of cybernetic devices are notably analogue, relying on continuous electrical signals or mechanical movements. Still, the ties between electrical technology, information transfer and biology were explicitly considered here, but not for the first time. Before stepping again towards contemporary artificial life, it is worth returning to the beginning of the 19th century, this time to understand how electricity first came to be associated with *life*.



Galvani set up numerous experiments to test the conditions under which electricity could be used to force the muscles of a deceased frog's legs to contract. The initial discovery of the phenomenon occurred by accident. One of Galvani's assistants touched a scalpel blade to a nerve in the frog's leg while it lay on a table shared with a machine designed to generate electric charge. To his surprise the leg twitched violently. To everybody's wonder, the experiment was repeatable.

Frog legs and animal electricity

Many readers will be familiar with Mary Shelley's 1818 tale *Frankenstein; or, The Modern Prometheus*. If not with the book (which is well worth reading!) then at least with the classic black and white 1931 film starring Boris Karloff as the monster. Frankenstein's monster was cobbled together from parts of corpses. It was animated, in a way that the narrator of the tale keeps secret, with electricity.

Shelley was inspired to write this story, warning of the potential dangers of artificial life, by the discoveries of the Italian physician Luigi Galvani. In 1771 Galvani had discovered that electricity caused a dead frog's leg to twitch. Although Galvani's ideas about bio-electricity were largely superseded by those of his contemporary, Volta, he nevertheless set people wondering if electricity was life's long hidden secret key. In 1803, Galvani's nephew, Giovanni Aldini, aimed to determine if electricity might be an elixir to reanimate victims of drowning. He recovered the body of the murderer George Foster from the gallows and had it carted to the Royal College of Surgeons. Before an audience, he applied electricity to the corpse. Its face quivered, an eye opened, its fist clenched and punched. The corpse's

legs kicked and its back arched in a gruesome display. But it didn't return to life. That was left to Frankenstein's monster in Shelley's purely fictional account of a (scientific) obsession carried too far.

Robotics

The term *robot* appears to have been born through the pen of Karel Čapek in his short play, R.U.R. (Rossum's Universal Robots). First published in 1921, this literary work explores the use of intelligent robots as slave labour, and their revolt against the human masters, their creators, who enslave them. In the first feature length science fiction film, the silent *Metropolis* (Fritz Lang, 1927), the role of electricity in animating a woman-robot-cyborg is made apparent by visual effects that instigated an entire genre of cinema. In this tale human workers form an underclass of what are essentially biological automata, slaves to wealthy masters who own the factories in which the workers must spend their dark, mechanical days.

Original movie poster for *Metropolis* (Fritz Lang, 1927).

It didn't take long before robots left the pages and screens of science fiction and entered the physical environment. With the immense power of the machines of the Industrial Age and control systems devised since the middle of the 20th century, complex industrial robots could become a reality. Manufacturing and industry have never been the same since. But pleasingly, not all robots are built with productivity in mind. Many have been assembled to test the bounds of human ingenuity, to probe questions that relate to our understanding of intelligence, the possibilities for machine creativity, or to help us comprehend biology and ecology. Many, as most readers will be aware, have been built as artificial life just for play.

A note

If there is one section in this brief technological history that deserves to be expanded, this is it. The ideas raised within Cybernetics and robotics are tightly intertwined with those of contemporary Artificial Life. But rather than dwell on these topics in this brief introductory section, the extent to which they impact on current thoughts will be clarified throughout the following chapters.

Further reading

Ashby, W. R. (1952). "Design for a brain, the origin of adaptive behaviour". New York.

Galvani, L. (1791/1953). "Commentary on the Effects of Electricity on Muscular Motion". Bologna, Italy / Norwalk, Connecticut, Burndy Library.

Ihnatowicz, E. (1986). "Cybernetic art: a personal statement", Edward Ihnatowicz.

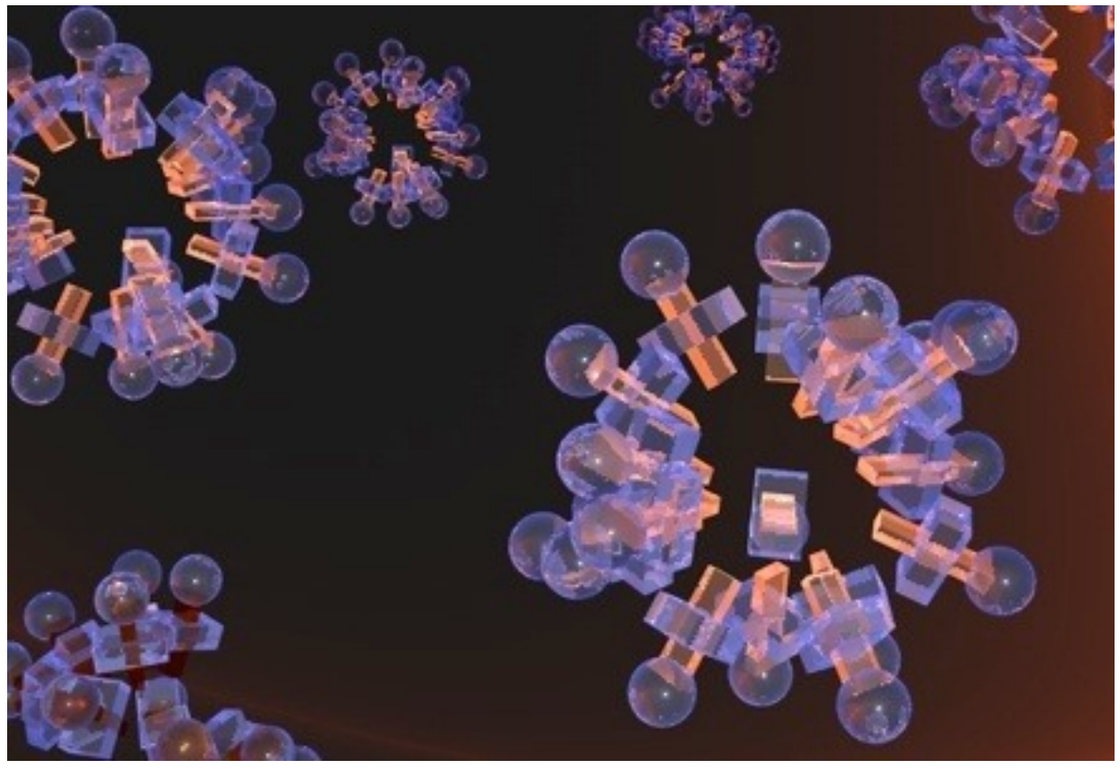
Reichardt, J., Ed. (1968). “Cybernetic Serendipity: the computer and the arts”. London, UK, Studio International.

Shelley, M. (1818). "Frankenstein or the modern Prometheus", Oxford University Press.

Chapter 2

Artificial Chemistry

Evolution by natural selection has built an extensive range of organism bodies. These are chemical systems for hosting chemical replicators to protect them, to find energy to maintain a functional outer capable of collecting resources for them, and to orchestrate mate selection and breeding for the transferral of them to create new bodies. Beneath it all is chemistry. This chapter considers simulations of the emergence of life from molecular building blocks through reaction-like processes.



SECTION 1

The basic elements

TOPICS

1. Alchemy, a chemistry of life
2. The Miller-Urey experiment
3. What is Artificial Chemistry?
4. Fact free science

An alchemist's workshop. Detail from the title page of an early book on mining: Beschreibung allerfürnemisten mineralischen Ertzt unnd Bergkwercks Arten. Lazarus Ercker, d. 1594. Frankfurt am Mayn– Imprint Frankfurt am Mayn: Joannem Schmidt in Verlegung Sigmundt Feyerabends, 1580.

The point of view emphasized in this contribution sees adaptive systems as based on combinatorial objects of a special kind. Objects, namely, that support interaction patterns whose targets are the objects themselves. Systems containing a self-referential constructive loop represented by objects that define functions that in turn modify objects are strong candidates for complex behavior, because the combinatorial explosion at the object level carries over to the level of interactions. [...] Chemistry is a prime example. Molecules carry reactive properties by which they modify other molecules in specific ways, thus creating new molecules with new interactive characteristics – W. Fontana, 1990.

Alchemy, a chemistry of life

As discussed when considering autopoiesis as a theory of living systems, organisms are a special kind of chemical machine. They can be viewed as self-sustaining networks of processes of transformation and destruction of molecules. Chemistry is at the heart of life. It is a discipline with a long past rooted in convoluted and idiosyncratic rituals and experiments with the materials of alchemy. Many alchemical and chemical procedures have been highly relevant to Artificial Life throughout the ages. Alchemy's origins date back to ancient Egypt. The diverse practices that characterise it continued through classical Greece and Rome, into the Medieval period and the Renaissance in Europe, the Islamic empire, India and China, and were common until just a couple of hundred years ago. Amongst alchemy's many activities was the manipulation of matter, particularly the transformation of metals

from the most “base”, lead, into more “noble” elements like silver and gold. Tied to this concern was the alchemists’ obsession with unlocking the secret of life, even to go so far as to generate humans in glass vessels. The manufacture of an elixir promising eternal youth was also on the agenda. The alchemists used magical incantations and invoked spiritual references, legends and myths that are out of place in today’s laboratories, but they also developed many practices and pieces of equipment that are still essential in modern labs. Researchers concerned with biochemistry, in particular with the the production of “wet” Artificial Life (e.g. the manufacture of a protocell from the same building materials as nature), have particular reason to be grateful to the alchemists of days past. Many who are more interested in software-based artificial life also attend to Chemistry by engaging in a sub-discipline known as “Artificial Chemistry”. This is arguably an essential bridge between simulation experiments and tangible, synthetic biological organisms. Amusingly, some software-based researchers have explicitly acknowledged their fascination with alchemy: Walter Fontana has devised a language dubbed, *AlChemy* and Hiroki Sayama calls users of his Swarm Chemistry tools *alchemists*!

The Miller-Urey experiment

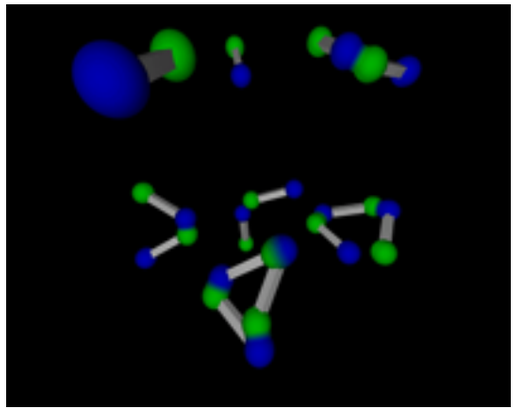
In the early 1950s, Stanley Miller and Harold Urey, two American chemists, devised a laboratory apparatus resembling something an alchemist might have used with an aim quite in keeping with those of some alchemists; to create the building blocks of life from inanimate matter. Their apparatus consisted of a tubular glass circuit passing through two chambers, one filled with liquid water modelling an ocean, the other filled with gases – water vapour, methane, ammonia, and hydrogen – thought to have been present in Earth’s early atmosphere. The mini-ocean was heated to produce vapour which, mixed with the gases, travelled around the loop into the mini-atmosphere flask. Here the entire gas mixture passed between two sparking electrodes to mimic the effect of lightning. The gases were then re-condensed and looped back into the mini-ocean vessel. The purpose of this cycle was to test a hypothesis that conditions on the prebiotic Earth were suitable for generating complex organic compounds. After a day in circulation the solution had already turned pink. In a week it was deep red and cloudy. When the resulting substance was analysed, amino acids, the biological building blocks of proteins were identified, having assembled themselves under the experimental conditions.

What is Artificial Chemistry?

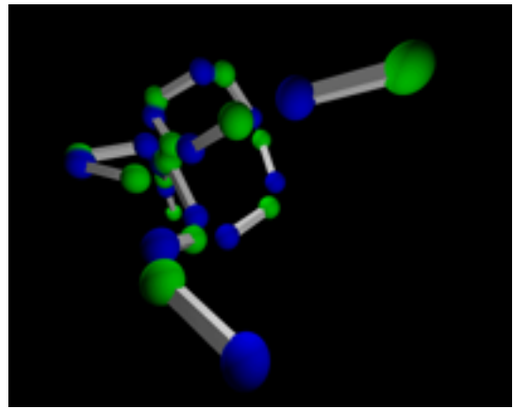
The Miller-Urey experimental approach is similar to that taken by some Artificial Life researchers engaged in Artificial Chemistry. The hope is that if some well-considered (virtual) ingredients are mixed together in an appropriate way and allowed to interact with one another according to well-considered interaction rules in a suitable (virtual) environment, they will self-assemble or self-organise into more complex structures. Hopefully the process can inform our understanding of the emergence, behaviour and evolution of life. Together, a researcher's specification of these three aspects of a simulation: atomic or molecular ingredients, reaction rules, and the dynamic process by which they interact in a particular virtual environment, define an artificial chemistry.

The virtual elements of artificial chemistries are analogous to atoms and molecules but they may be represented in many different ways. Possibilities include physically-modelled spheres or polyhedra that bond with one another based on collisions occurring as they move in 3D space and change state based on specified transition rules. For instance consider Self-Organizing (solid) Cellular Automata (Gallery 2.1). This system was so named because the state of a bond changes in response to the proximity of other bonding sites according to a transition table like those found in Cellular Automata – we cover Cellular Automata later.

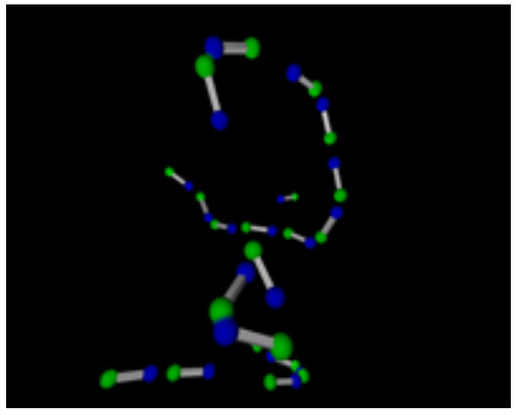
Two-dimensional artificial chemistries are also useful, their elements might be polygons or line segments with bonding



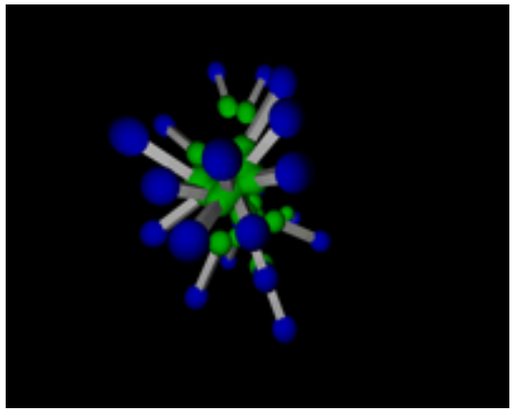
These structures are generated using an artificial chemistry, *Self-Organizing Cellular Automata*, by the author. Here, short, stable chains and triangles self-assembled under conditions where green and blue bonding sites at the ends of otherwise inert polyhedra attract one another.



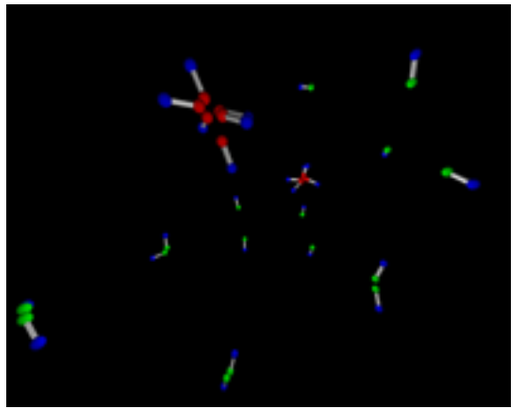
A long but tangled self-assembled chain. This is generated under the same conditions as the previous formations with green and blue bonding sites attractive to one another but in this case, like colours also slightly repel one another.



Untangled self-assembled chains. These are generated under the same conditions as the previous formations with green and blue bonding sites attractive to one another but in this case, like colours repel one another relatively strongly.



A self-assembled cluster. These structures self-assemble when green sites are set to attract their own kind and blue sites to repel their own kind.



Clusters generated by fracture of oversize green-centered clusters. The red-centered clusters were previously a single green-centered cluster. When a threshold of self-attracting green is surpassed within a structure, a transition of the sites occurs from green to red causing a split in the cluster.

sites at their vertices or along edges. In some artificial chemistries alphanumeric characters, binary digits or other symbols take the place of atoms. These can be combined, transformed and dissociated based on simple deterministic or probabilistic rules. Their motion might not be explicitly modelled, instead each could be probabilistically assigned a set of neighbours for a particular step of time in a simulation. The set of neighbours might include all other molecules in the simulation, or a subset based on their position on a lattice. Many simulations have such **stochastic** components.

Not all artificial chemistries explicitly model individual atoms and molecules. It is also possible to represent the presence of a particular substance in a simulation by its concentration. The simulation then takes the form of a set of equations describing the rates at which chemicals are converted from one type to another.

Like the Miller-Urey experiment, artificial chemistries might tell us how likely it is that certain biologically relevant (real) molecules appear under particular conditions. But very often the idea is to abstract away most of the detail of real biochemistry, reducing it to a comparatively simple computational system. In this way general principles of self-assembly, self-organisation, self-maintenance, self-replication and the emergence of complexity from basic building blocks can be studied, without reference to particular biomolecules.

Fact free science

Not all researchers are convinced by this abstract approach in Artificial Chemistry, or in Artificial Life broadly. Artificial Chemistry seems to some to be as bizarre and perhaps as worthless as alchemy is now perceived to be. Artificial Life practitioners have been subject to the accusation (originally made by biologist John Maynard Smith) that they are conducting "fact free science". Such criticism reminds us of the need to check our assumptions and validate our models, before drawing conclusions about (biological) reality based on (Artificial Life) simulations. A properly validated simulation is a powerful tool for understanding the real world, computational chemists know this well. But in and of themselves, abstract computational systems are *interesting*, regardless of their ties to the physicochemical world. New, artificial systems can be worthy of study in their own right. They make their own facts.

The abstract study of "self-governed" processes is particularly important in light of Maturana and Varela's focus on the self-production of living machines. But self-assembly and self-organising processes are also important because they provide the only mechanism by which nature can create complexity be-

fore an evolutionary process commences. The processes must be understood in order to explain how the first replicating molecules capable of evolution came into being. This is an important reason for studying artificial chemistries.

A later section of this text is completely devoted to virtual ecosystems that model interactions between indivisible organisms, and some of the research questions they can be used to tackle. Organisms in such simulations are atomic, hard-coded units of the simulation rather than phenomena emergent from chemistry. But in the current chapter we will describe models to demonstrate how an artificial chemistry might in fact *build* organisms from lower level building blocks.

Further reading

Miller, S. L. (1953). "A production of amino acids under possible primitive Earth conditions." Science 117(3046): 528-529.

Dittrich, P., J. Ziegler and W. Banzhaf (2001). "Artificial Chemistries – A Review." Artificial Life 7(3): 225-276.

Dorin, A. (1998). "Physically Based, Self-Organizing Cellular Automata", in Multi-Agent Systems - Theories, Languages and Applications, Zhang C., Lukose D. (eds), LNAI 1544, Springer-Verlag, pp. 74-87.

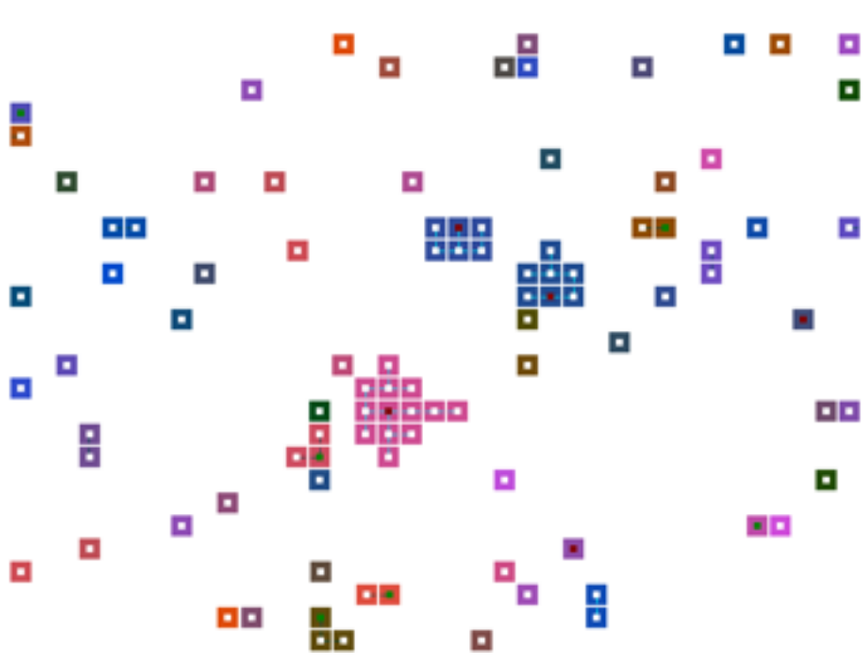
Fontana, W. (1990). “Algorithmic Chemistry: A Model for Functional Self-Organization”, SFI working paper, Santa Fe Institute. 1990-011. Technical report LA-UR-90-1959.

SECTION 2

Chemistry before evolution

TOPICS

1. Space and atoms
2. The simulation time step
3. Chemical reactions
4. Organisms



A screenshot from an artificial chemistry capable of supporting simple virtual ecosystems. Coloured squares represent atoms distributed on a grid. The cyan lines between them indicate chemical bonds. Squares filled with red act as enzymes to break chemical bonds. Squares filled with green act as a chlorophyll-like catalyst. Details of this model are described in the current section.

The defining properties of ecosystems are the exchanges and transformations of energy and matter between (and by) processes in the abiotic environment and the species inhabiting it. Likewise we have noted that the transformation of molecules within organism bodies is one of *their* defining properties. Therefore there are sound reasons for wanting to build an ecosystem model from the chemical level up, even though not all research questions demand it. In principle at least this task should be achievable. In practice, it is quite difficult.

Here we will describe one artificial chemistry system in detail to illustrate an approach to their construction, and to demonstrate that something like the kinds of interactions between organisms seen in nature might be realised bottom-up. In this system, designed by the author with computer scientist and philosopher Kevin Korb, we take a mid-level model of chemistry concerned with energy acquisition, storage and transformation into biomass. The model does not implement self-reproduction, evolution or the self-assembly of organisms. It is much simpler than that. Still, it demonstrates the feasibility of synthesising an entire ecosystem of interacting organisms, and their abiotic environment, from the same basic building blocks.

Space and atoms

The artificial chemistry is based upon a set of two dimensional, mobile, nonintersecting, square “atoms” on a lattice. Atoms may bond to neighbours in their **von Neumann neighbourhood** on the lattice by sharing virtual electrons

across their edges according to the rules of a virtual chemistry of covalent bonding: Each atom type has a set of electrons in one or more shells. The number of electrons and the fullness of an atom's outer shell determine the bonds in which the atom can participate. In all cases, some energy threshold is required to initiate (or break) a bond, and the result of bonding (or breaking a bond) is either the capture of an amount of energy or its release. Additionally, for each type of bond, parameters of the simulation determine the probability of specific bonds forming or breaking given the availability of the requisite energy.

A **catalyst** is said to be present at a reaction site when an atom involved in the reaction neighbours an atom of a designated catalyst-type. To support the existence of the virtual organisms that are intended to appear, four types of catalyst are required. A chlorophyll-like catalyst is needed that, in the presence of "sunlight" (described shortly), manufactures a complex molecule equivalent to sugar. An enzyme that breaks down this sugar, releasing the chemical energy stored in its bond is also needed. For simplicity and clarity, separate enzymes that decompose "organic" bonds that are not sugar and "inorganic" bonds are included too.

Energy that is released during a reaction is distributed throughout any continuous atomic structure that contacts directly or indirectly (through intermediate neighbours) the reaction site. This energy is available for making or breaking chemical bonds by any atoms that receive it.

The simulation time step

The simulation progresses in discrete time-steps. At each step, for each atom, it must be determined stochastically whether each bond should break or join based on the site-types, the presence of catalysts, the availability of energy and the bonding probabilities. Energy released during a reaction is totalled in each neighbourhood of connected atoms for use in that time-step by reactions that absorb energy. A reaction that requires energy to be expended can only occur if the neighbourhood of atoms involved has accumulated sufficient energy. Reactions occur in random order, consuming or contributing energy to and from the total amount available in their neighbourhood. A reaction that requires more energy than is available cannot proceed.

Energy released from a chemical bond must be used in that time-step or it is released in non-recoverable form. The only way energy can be stored is in complex molecules.

Sunlight is incident on all atoms at a rate governed by a parameterized sine function for use by the chlorophyll-like catalyst during photosynthesis.

In addition to stochastically determined bonding, within a time-step atoms may be moved one square on the grid in a random direction or they may remain stationary. Bonded atoms (forming molecules) are moved identically to preserve their topology. Collisions are not permitted.

Chemical reactions

This simple model supports the existence of autotrophs (creatures that make their own food by harnessing energy from the sun or existing chemicals) and heterotrophs (creatures that eat other creatures), including decomposers.

The abiotic environment is made of the same molecules and atoms as the simulation's organisms. The bond structure and context enables us to label molecules as inorganic or organic, as components of a metabolic system or as abiotic. Thus the "abiotic environment" is just the set of atoms and molecules that are not bonded to a structure identified as an organism.

The abiotic environment consists of virtual atoms from the set {A, B, C, O}. Atoms may also be enzymes for sugar decomposition (break A-B bonds), biomass decomposition (break C-C bonds) or chlorophyll for sugar construction (make A-B bonds, break A-O and B-O bonds). The probabilities for these significant reactions are given in the included reaction table.

Bond energy (right hand column of the table) must be supplied to break a specified bond and is released when the bond is made. Negative bond energy values indicate a release of energy when a bond breaks and energy must be supplied to make these bonds.

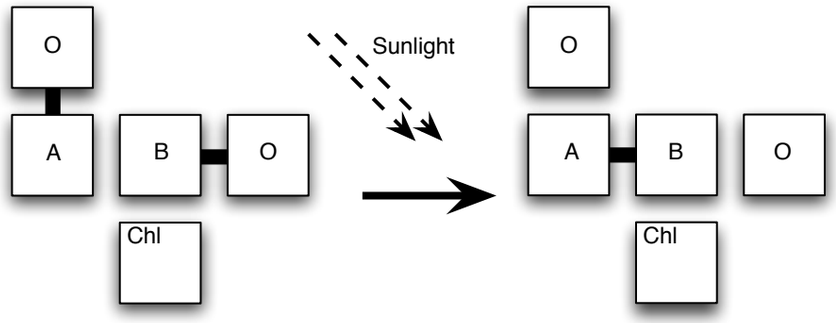
In order to sustain the required organism forms, several reactions must be supported. These are illustrated below. Only bonding possibilities of relevance to the discussion are shown, even though other bonds may also be supported. A line between tiles in the diagrams indicates a bond between the atoms they represent. Catalysts are labelled Enz (enzyme) or more specifically Chl (chlorophyll).

Photosynthesis constructs sugar molecules from water and carbon dioxide, releasing oxygen and water. It requires the presence of the catalyst chlorophyll and incident sunlight. In our virtual chemistry the reaction is represented abstractly as:

AO + BO —(chlorophyll & sunlight)⟶ AB + 2O

BOND	MAKE PROBABILITY	BREAK PROBABILITY	MAKE PROBABILITY (CATALYST)	BREAK PROBABILITY (CATALYST)	BOND ENERGY
A-B	low	low	high (chl)	high (enzAB)	-high
C-C	moderate	low		high (enzCC)	-low
A-O	high	low		high (chl, enzAO)	+low
B-O	high	low		high (chl, enzBO)	+low
C-O	low	moderate	high (enzCC)		+low

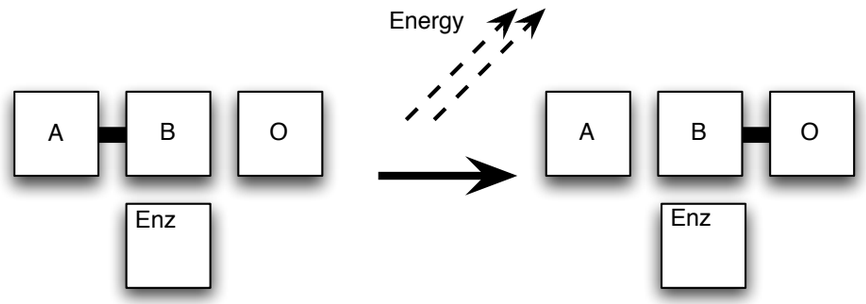
Reactants A and B may be presented to the catalyst bonded to O atoms or free. The process of photosynthesis may be represented in many potential atom arrangements.



Respiration. Real sugar molecules may be broken down in a process of respiration to release energy. The process utilizes oxygen and an enzyme to break down the sugar. Carbon dioxide and water are released. In our virtual chemistry the reaction is represented abstractly:

O + AB —(enzyme)⟶ A + BO + energy

The process of respiration may be represented in many potential atom arrangements.

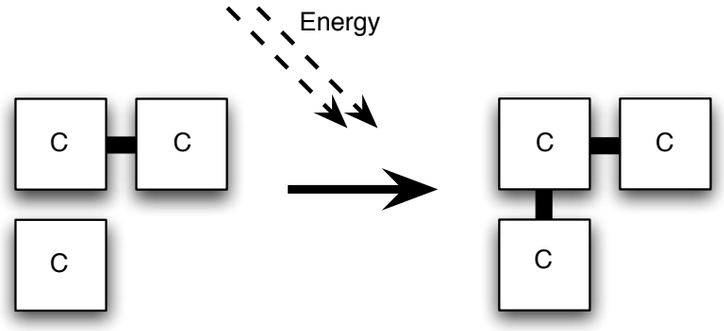


Biosynthesis. Organisms are able to add to their biomass by constructing bio-molecules from those they ingest. Growth occurs when a structure binds to atoms employing a reaction that requires energy. These bonds would not normally form in the absence of the energy required to create them (or the presence of a suitable catalyst). Such bonds may also break down spontaneously with probabilities as indicated in the reaction table. Hence an "organic" structure formed of these bonds must produce sufficient energy to sustain itself against natural decay by rebuilding broken bonds and by adding new material in a breach.

When an organic bond is broken energy is released into the neighbouring structure. The amount that may be captured by a particular neighbouring atom will be insufficient to remake the bond instantaneously without an additional energy source. In our virtual chemistry the biosynthesis reaction is represented abstractly:

C + C —(energy)⟶ C2

The process of biosynthesis may be represented in many potential atom arrangements.



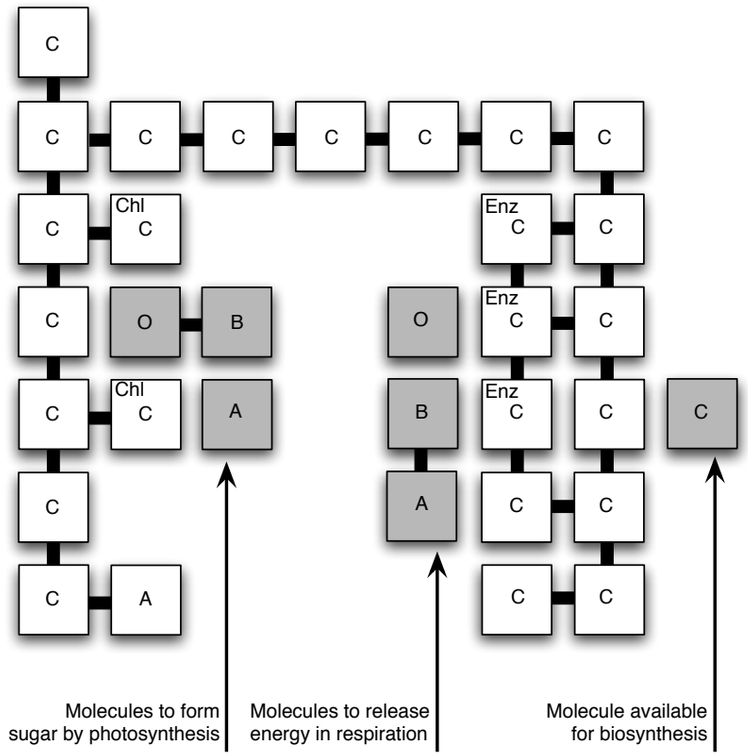
Organisms

Many different atom configurations could conceivably fill the role of organisms in a virtual ecosystem built from these atoms. For example, a chemosynthetic autotroph that generates its energy from inorganic molecules without the need for sunlight might be designed. In the chemical system described, one way to achieve this is by having a configuration that obtains free O atoms and binds them to available A or B atoms. Given the natural affinity of A and B for O in the model, suitable atoms may be scarce unless a catalyst is employed to split A-O and B-O. The elements may then rejoin against the surface of the structure. In this case almost any structure containing a suitable catalyst on its surface would suffice in the role.

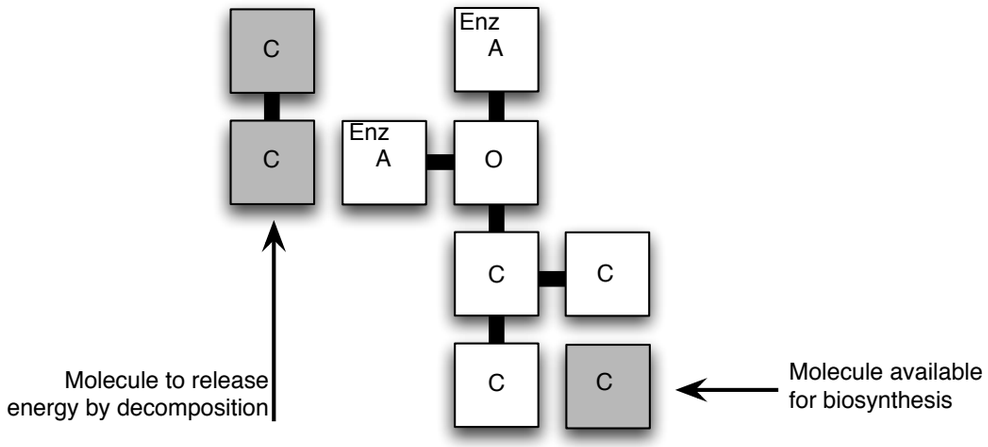
For illustrative purposes, two other organisms are shown and explained (see figure captions) here.

Photosynthetic autotroph. This photosynthetic autotroph harnesses chlorophyll to manufacture sugar. It also holds an enzyme to decompose sugar and a vacuole (cavity) in which to manufacture and store it. Countless variations on the design are possible. They needn't capture the sugar molecules as long as they maintain sufficient sugar concentration in their vicinity.

In the design presented, one internal wall of the vacuole anchors chlorophyll. Any A-O and B-O molecules that contact it will be converted into sugar but remain trapped. By chance, an A-B molecule may later touch the opposite enzyme-laced wall where it will participate in respiration, releasing energy through the structure and allowing biosynthesis.



Decomposer. A heterotroph in the present model could break down A-B sugars produced by other organisms using an enzyme. Alternatively, a heterotroph can act as a decomposer (illustrated) if it possesses a catalyst to break down the C-C bonds of another structure. This catalyst must be isolated from its own organic C-C structure. One way to achieve this is with an intermediate O atom bonded to C with the aid of a catalyst.



Summary

This simple virtual chemistry permits many types of interaction to occur between its organisms, but it lacks the self-organising forces that automatically generate interesting structures, everything in this model must be specified manually. The simulation is also unhelpful for the study of evolutionary processes since it is limited by its present lack of a mechanism for encoding reproduction with variation. Hence, while it is a useful demonstration of the possibilities for virtual ecosystems built using artificial chemistry, it really only takes an enticing first step. Higher level simulations that abstract away some of the atom-level detail in the model just described, while adding facilities for reproduction and evolution, have also been constructed. Some of these, including *Tierra*, a world built of atomic programming-language instructions, are described in a later section on virtual ecosystems. The next section describes an artificial chemistry with the physicality of the tile-based model just explored, but this is linked to a mechanism for transferral of information between elements to allow evolutionary processes to occur.

Further reading

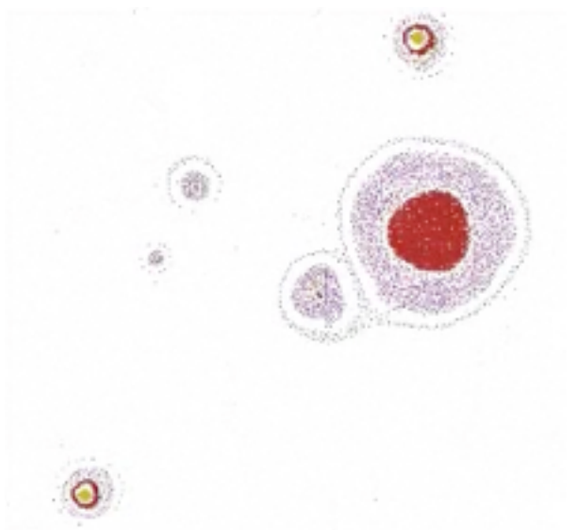
Dorin, A. and K. Korb (2007). "Building Artificial Ecosystems from Artificial Chemistry", 9th European Conference on Artificial Life, Lisbon, Springer-Verlag, pp. 103-112.

SECTION 3

Imaginary chemistries

TOPICS

1. Why build imaginary chemistries?
2. Swarm chemistry
3. Reactions
4. Evolution



Some cell-like structures formed within an evolving Swarm Chemistry simulation. © Image copyright Hiroki Sayama. Used with permission.

Why build imaginary chemistries?

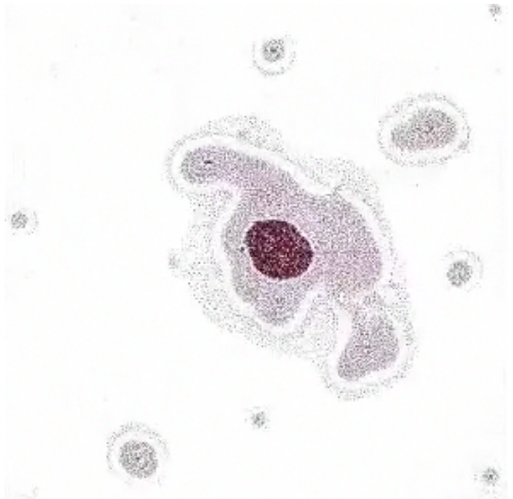
Artificial Chemistry, as it is practiced within Artificial Life, is quite different to that practiced within Computational Chemistry. The computational chemist uses tools such as Molecular Dynamics simulation to understand and predict how real atoms and molecules will behave under particular conditions. The Artificial Life researcher on the other hand, is free to experiment with invented interaction rules and novel pseudo-molecular behaviours. The artificial chemistry described in the previous section is loosely based on the real biological processes of energy acquisition, its storage and release, and the construction of biomass. But rebukes about "fact-free science" aside, artificial chemistries allow us to study the general principles of dynamical system construction beyond what we find in nature. With general knowledge gleaned from imaginary chemistries, we may learn to synthesise new systems that self-assemble into complex structures and repair damage to themselves, we may derive new medical technologies that assist an organism to maintain itself when injured, to repel the invasion of bacteria, eliminate a viral infection or adapt deeply and dynamically to variations in its environment.

Self-assembly, self-repair and self-(re)organisation are valuable engineering principles for the creation of new technology. The Swarm Chemistries originated by Hiroki Sayama occupy this experimental space. Despite their apparent simplicity, they are capable of fascinating dynamic behaviour that may one day inform new technological innovations beyond the

screen. In particular the mechanisms of Swarm Chemistry suggest coordination strategies for swarms of mobile robots.

What is Swarm Chemistry?

Swarm Chemistry is only loosely speaking an artificial chemistry. Its elements behave more like the mobile *boids* of Reynolds' flocking software, a system we describe in detail later. Each element can perceive others in its vicinity and responds to their presence. Each element can move through a continuous 2 or 3D space with its own set of parameters governing steering towards or away from other elements, alignment with the direction of travel of neighbours, avoidance of collisions with them or just random wandering. The parameters are specified as: a radius of local perception, a default speed, a maximum speed, the strengths of cohesive, alignment and separating forces between an element and its neighbours, the probability of random steering and the tendency to travel at the default speed.



Organism-like forms emerge from an evolving swarm chemistry simulation. © Copyright Hiroki Sayama. Used with permission.

A swarm "species" consists of elements with identical movement parameters. The interesting behaviours Swarm Chemistry generates arise when several species mix in a single space. This mixing, and the ensuing emergent behaviour, is termed a "reaction" within the model. Parameters for many phenomena have been successfully discovered by the alchemists (users) of the system.

Reactions

Spontaneous segregation appears readily within the Swarm Chemistry framework. Agents spontaneously cluster with others of the same species when the differences between species' parameters ensures that the local environmental conditions each generates and gravitates towards differs.

Reactions may produce macro-level multi-species movements that arise when one species tends to want to approach another while the parameters of the species being approached have it preferring to avoid the encounter. The net result is a mass of particles of the two species chasing each other around the screen while tending to stay in proximity with their own kind.

In some cases, one chemical species forms a dynamic membrane around a central core of members of a different species. Sometimes the confined species rotates or oscillates *en masse* within its container.

Evolution

An inheritance mechanism has been encoded within later versions of Swarm Chemistry. Active elements in this version of the software contain multiple sets of parameters, one of which dictates element behaviour at any time. Upon collision with

another element, the parameter sets of one element are transferred to the other participant in the collision, possibly with some mutation that changes the values being copied. One of the received parameter sets is randomly chosen to become active within the target element that now enacts its motion rules based on this. The transfer of information between elements allows an evolutionary process to become established. Sayama has discovered the emergence of collections of particles capable of behaviours visually and conceptually reminiscent of reproduction and various ecological phenomena.

Summary

Swarm Chemistry and the tile-based ecosystem given previously have been selected for discussion because their outcomes can be interpreted visually. It is therefore hoped that they serve as an accessible introduction to Artificial Chemistry and its potential to generate complete ecosystems. There are many other ventures in this field worth studying, not all of them visual. Two different systems of note include Wolfgang Banzhaf's self-organising binary strings (and later work he did with Peter Dittrich), and Tim Hutton's self-reproducing, membrane-enclosed cells.

Artificial Chemistry encompasses a broad range of approaches. As can be seen from the examples given, these models are often described in terms of metaphors adopted not just from Chemistry but from as far afield as flocking, cellular automata and ecosystem modelling. Seemingly any system of interacting dynamic components might be classified under the banner of Artificial Chemistry! Since even completely imaginary chemistries can be informative, as long as the metaphors used are helpful to describe or explain observations of these dynamical systems, and their authors are clear that these are *metaphors* only, there is, in this author's opinion, no harm in using them.

Further reading

Sayama, H. (2009). "Swarm Chemistry." *Artificial Life* 15(1): 105-114.

Sayama, H. (2011). Seeking open-ended evolution in Swarm Chemistry, In 2011 IEEE Symposium on Artificial Life. Nehaniv, *et al* (eds). Paris, IEEE: 186-193.

Banzhaf, W. (1994). Self-organization in a system of binary strings. Artificial Life IV, Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems. Brooks and Maes (eds), MIT Press: 109-118.

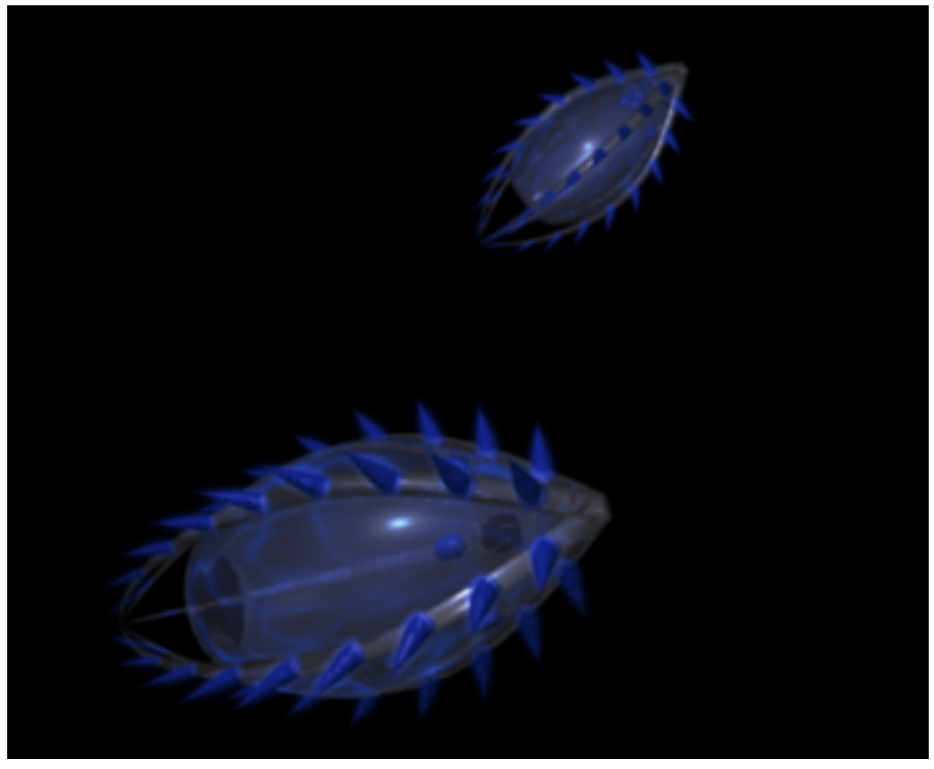
Dittrich, P. and W. Banzhaf (1998). "Self-Evolution in a Constructive Binary String System." *Artificial Life* 4(2): 203-220.

Hutton, T. J. (2007). "Evolvable Self-Reproducing Cells in a Two-Dimensional Artificial Chemistry." *Artificial Life* 13(1): 11-30.

Chapter 3

Artificial Cells

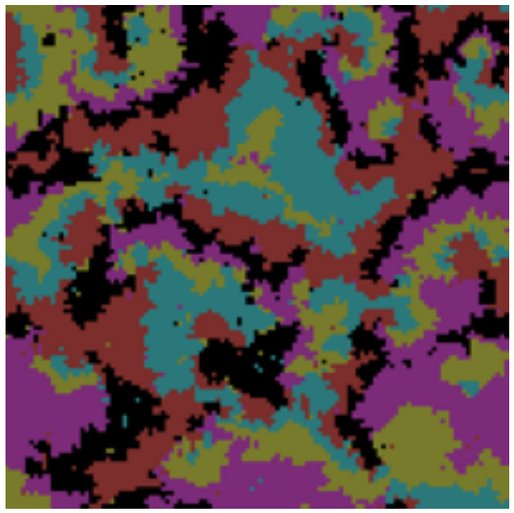
The interaction between individual biological cells gives rise to complete multicellular organisms. In this chapter we explore software systems that approximate the interaction of simple cell-like entities. Despite their simplicity, these are capable of generating surprisingly complex emergent patterns.

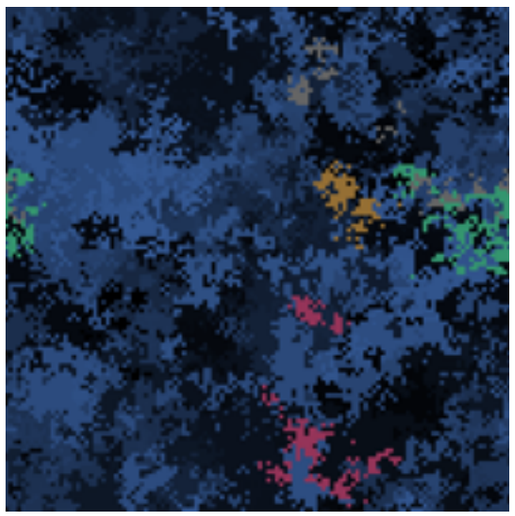


SECTION 1

Cellular Automata

TOPICS

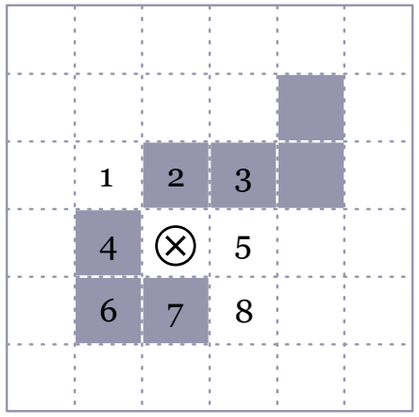




Cellular automata are convenient computational platforms for studying the *emergence* of large-scale patterns from local interactions. Here the patterns are visualised as coloured pixels in an array. Each pixel's colour represents the state of a machine connected to its neighbours in the grid. Hence, patches of colour indicate machines in identical states. Different rules for how machines change state will give rise to different kinds of large-scale patterns over time.

What are cellular automata?

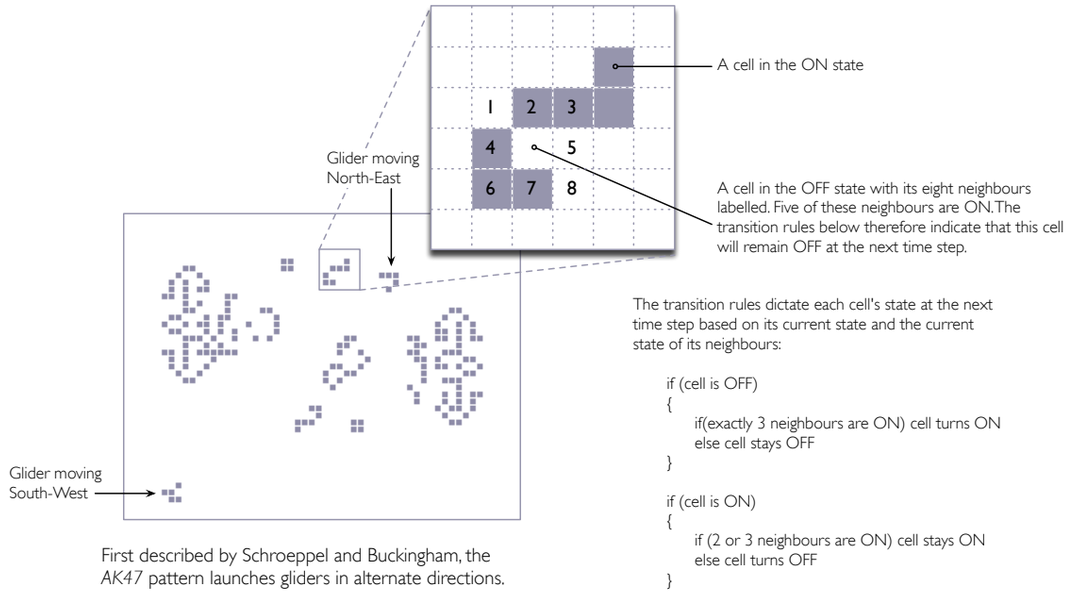
Cellular automata (CA) are finite state machines (FSMs) arranged on an infinite grid. Each machine (automaton) can be thought of as playing the role of a single cell, sometimes alone, sometimes as part of a multicellular structure. Each cell in the grid is connected directly to the FSM cells in neighbouring cells. All FSM states are updated in parallel at discrete time intervals. The transition each FSM makes at a time step is usually determined by a system-wide transition table. But the conditions that trigger an individual cell's change are based only on the cell's current state and the state of the neighbours to which it is directly connected. Some examples will be given below.



In this figure, a grid of FSMs is shown as a two dimensional array of squares. This is the usual way to visualise a CA. Each square in the figure, which shows a two-state CA, is coloured grey to indicate the FSM at that location is in the *on* state, or white to indicate the FSM at that location is in the *off* (quiescent) state.

The cell ⊗ is connected to cells 1-8 in its *Moore* neighbourhood. An alternative neighbourhood that is sometimes used in CAs is the *von Neumann* neighbourhood. For the cell marked ⊗ this includes only cells 2, 4, 5 and 7.

CAs are particularly interesting because, even though each FSM in the grid is connected only to its local neighbourhood, if the transition rules are carefully constructed, complex, dynamic, global behaviour nevertheless emerges across large regions of the grid as the FSM



Here are some patterns from the popular *Game of Life* CA. The transition rules are given at the right of the figure. Note the two gliders shown in the figure also.

states are updated. In fact, CAs are a quintessential example of *emergence* which, as already noted, is an important theme for contemporary Artificial Life.

The CA was introduced in the 1950s by John von Neumann at a suggestion from his friend, mathematician Stanislaw Ulam, concerning how to construct a self-reproducing computer program. He wished to know, "What kind of logical organization is sufficient for an automaton to be able to reproduce itself?"

This biological problem's solution has largely been beyond technological Artificial Life. But if a self-reproducing algorithm could be identified, then the process of self-replication could be shown to be possible within the world of formal machines. Prior to introducing the problem to the abstract, formal world of computation via CAs, von Neumann had envisaged a "kinematic" self-reproducing machine. The complexities of engineering such a system, and the lack of interest the practical problems held for von Neumann, meant the idea was superseded. Arguably the CA was his attempt to see a self-reproducing system realised in an environment where the problems to be solved were interesting to von Neumann the engineer, mathematician and theoretical computer scientist.

The formation of large-scale patterns (including self-reproducing ones) on a CA grid depends on the transition rules for the cells. Typically all cells in a CA grid share the same transition rule set. There must be a valid transition for each of the possible combinations of the current state of each FSM and the state of all its connected neighbours for the CA to be valid. CAs usually have a *quiescent state* that is sometimes referred to as a *blank state*. A cell in this state will remain in it if all of its neighbouring cells are also quiescent.

Apart from self-reproduction, emergent CA patterns can include various patches of same-state machines: spirals, spots, splotches, waves, circles, squares and diamonds. These are often dynamic. Their locations and boundaries change with every passing time step as cell states are updated. But more interestingly, dynamic structures that maintain a coherent, recognisable form as they drift across the CA grid can also appear. These cyclic moving patterns are called *gliders*. Structures that repeatedly emit a stream of gliders can also be devised. These are called *glider guns*.

The properties of dynamical systems such as CAs have been loosely characterized by referring to the long term behaviour of the system:

- *Limit Point* – system halts
- *Limit Cycle* – system falls into an infinite loop
- *Strange Attractor* – system behaves chaotically forever

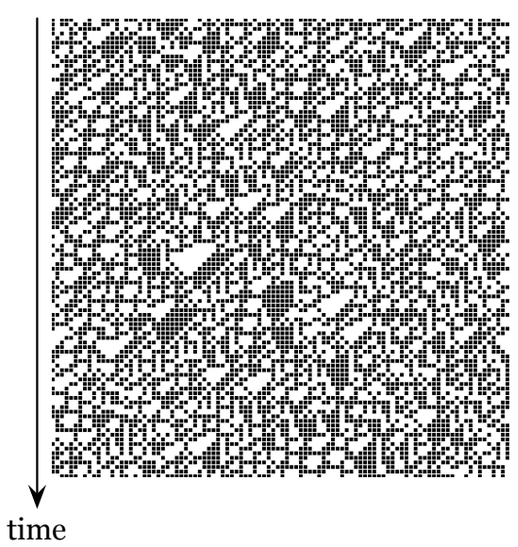
Computer scientist Stephen Wolfram has carefully studied 1D cellular automata in particular. These consist of a single row of connected finite state machines. A convenient way to visualise their dynamics is as a vertical sequence. Each row is set beneath its predecessors to depict the progression of all FSM states.

To relate the behaviour of CAs to the general principles of dynamical systems outlined above, Wolfram has classified their behaviour in four classes as follows.

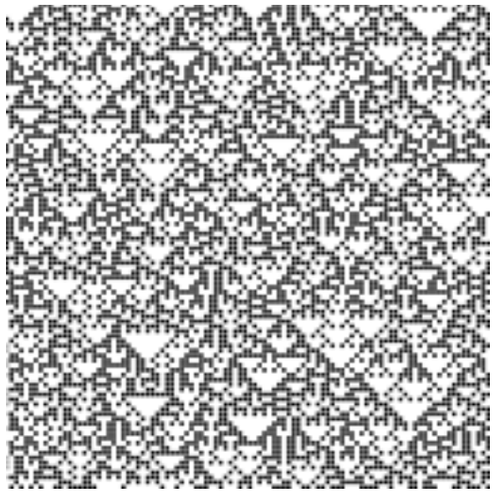
1. Reaches a homogeneous state (limit point)
2. Forms simple, separated, periodic structures (limit cycle)
3. Produces chaotic, aperiodic patterns (strange attractors)
4. Produces complex patterns of localised structures

The book, *Cellular Automata: A Discrete Universe*, by Andrew Ilachinski (World Scientific, 2001), analyses the behaviour of CAs in some detail and assesses the value of Wolfram's classifications.

We might wonder in what sense this last type of pattern is a form of digital artificial life. Certainly the same description seems to apply to organisms.



This 1D CA shows progression typical of Wolfram's class 3 dynamical systems – it produces chaotic, aperiodic patterns.



This 1D CA shows progression typical of Wolfram's class 4 dynamical systems – it produces complex localised structures that are neither completely chaotic, nor completely periodic.

The Game of Life [Try the interactive widget online]

The *Game of Life* is a CA rule set designed by mathematician John H. Conway and made popular in the early 1970s by Martin Gardner's *Scientific American* series *Mathematical Games*. Arguably this is the most famous CA of all.

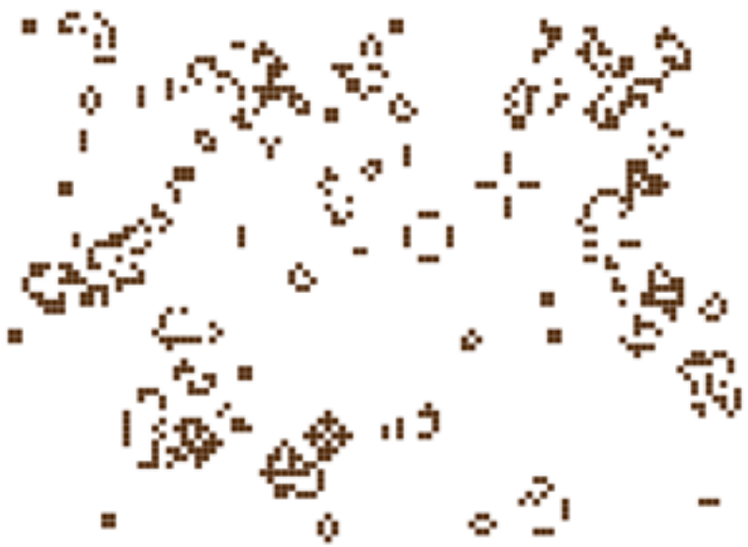
The Game of Life is a binary CA; each FSM can be in one of only two states. The CA employs the **Moore neighbourhood**. The system's rules are very simple.

Rules for the *Game of Life*

if (cell is OFF)
    if (exactly 3 neighbours are ON) cell turns ON
    else cell stays OFF

if (cell is ON)
    if (2 or 3 neighbours are ON) cell stays ON
    else cell turns OFF

Surprisingly many fascinating, dynamic patterns can be produced from this simple rule set. A complete Turing Machine has even been constructed by Paul Rendell entirely using the Game of Life: gliders carry messages from one location to another as the Turing Machine changes state or reads and writes to its cellular tape.



A snapshot of the *Game of Life*.

Langton's Loop from a simulation written by Tomoaki Suzudo.

Langton's Loop

Chris Langton, organiser of the first conferences specifically focussed on the field he dubbed *Artificial Life*, was a researcher into CAs. In 1984 he devised his own self-reproducing machine, *Langton's Loop*. Implicitly, the goal of such a system for Langton, like von Neumann, was for a multi-celled software structure to self-reproduce without explicitly calling a high-level method that adopts a global perspective of the operation. Hence the structure's self-reproduction must itself be emergent from the interactions of its components.

One way to achieve this would be to have a structure of two neighbouring cells, each which causes a quiescent neighbouring cell to adopt the same state. In this way, each active cell would independently make another. And then all four active cells could make eight, and so on. But this seems somehow trivial and boring. As problematic as it is to define clearly, von Neumann, and those who have taken up the gauntlet he threw down, have sought something "interesting".

Applications of cellular automata

CAs are more than fancy computer games. Some researchers have gone so far as to wonder if the entire universe is a kind of

CA. CAs also serve as a platform for philosophical thought experiments around the idea of emergence. Practical applications for the grid-based simulation method have arisen too. For example, cells can be used to represent quadrants of a landscape populated with forest, grassland and fire-breaks. Bushfire movements can then be simulated on the grid using transition rules such as one for cells that transition from unburnt, to burning, to consumed grass. A grass cell begins this sequence of transitions when a neighbouring cell enters the burning state. Other CA applications include understanding fluid dynamics, vehicular traffic or even modelling the interactions typical of complete natural ecosystems.

Further reading

Gardner, M. (1970). "The fantastic combinations of John Conway's new solitaire game *Life*." *Scientific American* 223: pp. 120-123.

Langton, C. G. (1984). "Self-reproduction in Cellular Automata." *Physica D: Nonlinear Phenomena* 10(1-2), pp. 135-144.

Langton, C. G. (1986). "Studying Artificial Life with Cellular Automata." *Physica D : Nonlinear Phenomena* 22(1-3), pp. 120-149.

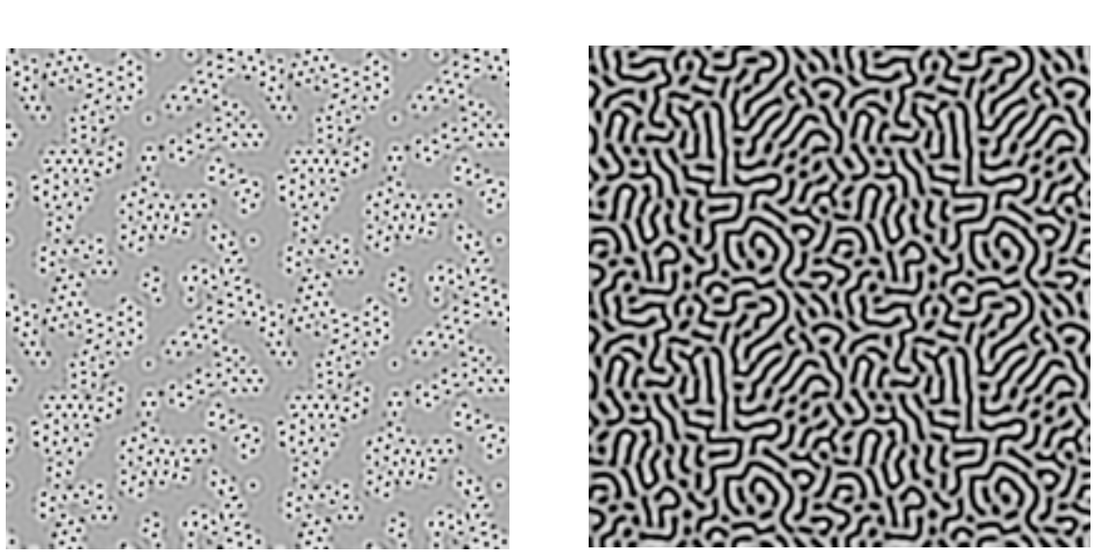
Wolfram, S. (1984). "Universality and Complexity in Cellular Automata." Physica 10D: 1-35.

SECTION 2

Reaction-Diffusion systems

TOPICS

1. What is a reaction-diffusion system?
2. Turing's reaction-diffusion system
3. Applications



Reaction-diffusion systems generate patterns reminiscent of those found throughout biology, especially on animals. In simulation, the patterns are usually rendered as coloured pixels in an array. Each pixel's colour represents the concentration of chemicals at its location in the grid.

What is a reaction-diffusion system?

In a reaction-diffusion (RD) system, chemicals are deposited onto a surface and diffuse across it over time. Might such systems be responsible for forming patterns during the development of an embryo. How? Naively we might expect that chemicals would always spread evenly across a surface until their concentration was uniform. However in RD systems uniformity is not necessarily a stable state, even when the system seems homogeneous to begin with. Small random variations can cause quite unexpected patterns to emerge.

Some patterns that seem typical of RD systems: a zebra's stripes, a giraffe's patches, fish markings and the spots of a leopard.

The chemicals that diffuse through organisms are called *morphogens* after their proposed role in biological morphogenesis. In 1952 the famous computer scientist Alan Turing, in what can only be considered a landmark paper, proposed that such systems could be modelled using partial differential equations. He felt RD systems might explain the development of whorled leaves, the

Alan Turing (1912-1954) was a pioneer, not just in computational models of morphogenesis, but in the theory of computation. He developed the *algorithm* concept formally, the idea of *computation* and an abstract, general purpose computing machine called the *Turing Machine*. Turing was also responsible for developing the *Turing Test*, a test of a machine intelligence deeply relevant for the philosophy and practice of the field of Artificial Intelligence.

Image: *Alan Turing*, by AIkon-1, a research project into art computing by Patrick Tresset and Frederic Fol Leymarie, hosted at Goldsmiths, University of London : www.aikon-gold.com. This portrait was drawn by a robot, AIkon-1, from a photograph. Used with permission.

pattern of tentacles on *Hydra*, gastrulation during the development of an embryo, dappled pattern formation and phyllotaxis.

A simple RD system

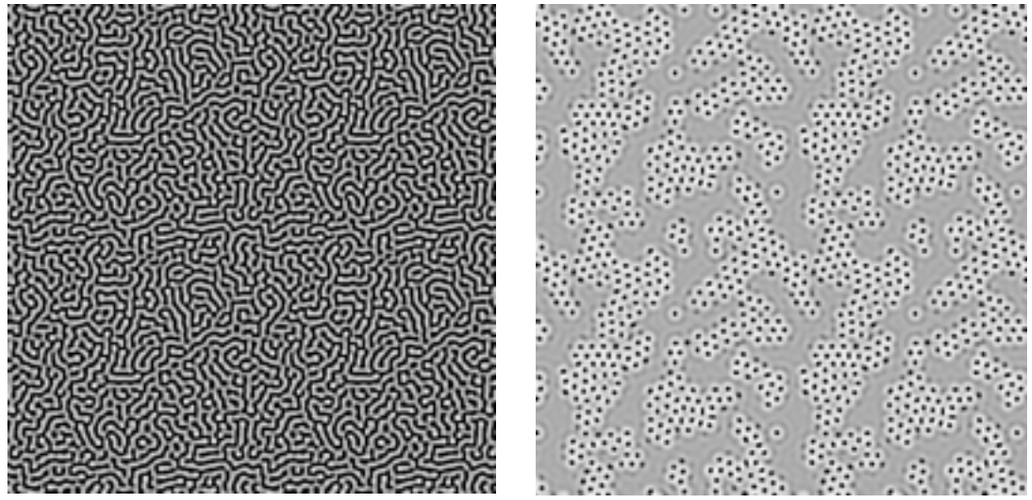
A simple RD system like those described by Turing has two chemicals, an activator with concentration at any point designated A, and an inhibitor of concentration I. These chemicals diffuse through space at different rates. Their change in concentration over time is described according to the following equations to be explained in plain English shortly.

$$\partial A/\partial t = F(A, I) + D_A \nabla^2 A$$

$$\partial I/\partial t = G(A, I) + D_I \nabla^2 I$$

The first equation describes the rate of change of concentration A, $\partial A/\partial t$. At any point this is influenced by: $F(A, I)$ which is a function F of the current concentrations A and I at that point; and by a measure of the rate of diffusion of A, the diffusion coefficient D_A; and by a measure of the concentration gradient at that point captured in the ∇^2 term. The second equation parallels the first but for the concentration of the inhibitor.

Above all else, it should be clear from this description that the two chemicals interact with one another through functions F and G.



Patterns generated from a simulation by Christopher G. Jennings.

It turns out to be possible to simulate the movement of chemicals using these equations on a grid. Changes in concentration within individual grid cells are computed by reference to the current concentrations in the cell, those of the neighbours, and according to the other terms in the equations given above. Here are the equations again, this time in terms of a discrete grid.

$$\Delta A_{i,j} = s(16 - A_{i,j} I_{i,j}) + D_A (A_{i+1,j} + A_{i-1,j} + A_{i,j+1} + A_{i,j-1} - 4A_{i,j})$$

$$\Delta I_{i,j} = s(A_{i,j} I_{i,j} - I_{i,j} - \beta_{i,j}) + D_I (I_{i+1,j} + I_{i-1,j} + I_{i,j+1} + I_{i,j-1} - 4I_{i,j})$$

The first equation states that the change of concentration of A at grid cell (i,j) is calculated by multiplying the constant *s* by a value computed by subtracting from 16 the current concentrations A and I at (i,j). This term corresponds to F(*A*,*I*) in the original equation given. The second term maintains the diffusion coefficient D_A from before and computes the Laplacian ∇^2 by summing the concentration *A* in the von Neumann neighbourhood of cell (i,j) – i.e. in its neighbours to the N, E, S and W – and subtracting 4 times the concentration at (i,j).

The second equation parallels the first but contains a function G different to F of the first equation. This function introduces β, a random value at every cell (i,j) that acts as a kind of *seed* over which irregularities in the pattern can form. This perturbation to homogeneity is introduced into the digital world to mimic nature's natural variation.

This basic RD system is capable of generating a variety of patterns as it stands. A visualisation of the grid consists in representing the concentration of the chemicals by colouring pixels. By varying the value of *s* different visual patterns emerge. Additionally, the system can be altered by having multiple diffusing and reacting chemicals. It is also productive to have one RD system lay down an initial pattern and then to run another simulation over the top of the first, using the first pattern as a non-random seed. Or part of the initial pattern can be frozen and the second simulation can be run in the unfrozen portion. It is even possible to use the initial pattern to non-uniformly vary the rate of diffusion of chemicals in the second simulation.

Reticulated (webbed) pattern on a giraffe. © Greg Turk 1991. Used with permission.

Applications of RD systems

The complex patterns RD systems produce have made them excellent sources of natural-looking textures for 3D graphics. In 1991, Greg Turk ran the RD simulations directly over the polygonal surfaces of 3D geometry. In this way he was able to generate coat patterns like those found on zebras and giraffes that were automatically tailored to the geometry of his models - they were generated directly over their surfaces. RD patterns have also been used as the basis for procedural techniques that generate 3D models. Where chemical concentrations reach a threshold for instance, new tentacles on a geometric structure supporting the simulation can be generated. By simulating the emission of chemicals from particle-like cells and the reaction of these particle-like cells to morphogens, it is even possible to generate two or three dimensional models from virtual cells – the subject of the next section.

Pseudo Zebra © Greg Turk 1991. Used with permission.

Further reading

Prusinkiewicz, P. (1993). "Visual Models of Morphogenesis." *Artificial Life* 1(1-2), pp. 61-74.

Turing A. M. (1952). "The Chemical Basis of Morphogenesis", Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences, Vol. 237, No. 641, pp. 37-72.

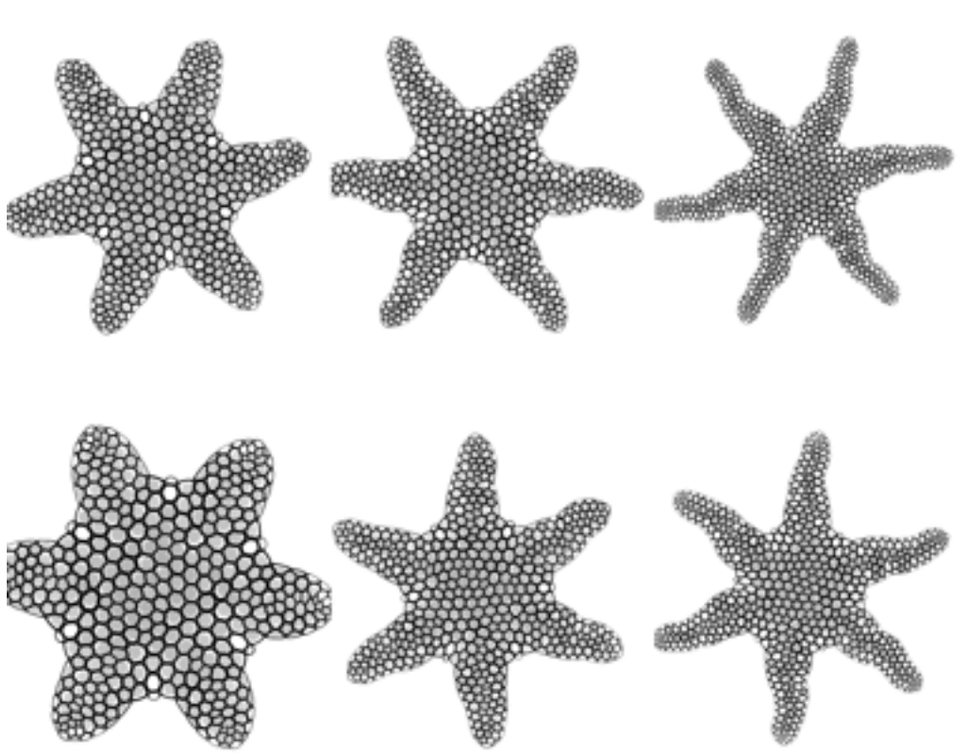
Turk, G. (1991). "Generating Textures on Arbitrary Surfaces Using Reaction-Diffusion", Computer Graphics, Vol. 25, No. 4 (SIGGRAPH '91), pp. 289-298.

SECTION 3

Mobile cells

TOPICS

1. The properties of virtual cells
2. Cellular textures
3. Cellular modelling



2D starfish grown using Ben Porter's SDS system.

Both reaction-diffusion systems and cellular automata are simulations based on cell-like entities in which the neighbourhood relationships that govern interactions are fixed and usually based on a regular grid. This needn't be the case. It is quite possible to simulate mobile cells that interact with one another as they come into range – that is, the neighbourhood relationships between cells are dynamic. This allows structures to form and disintegrate, much like they do in artificial chemistry simulations. In this case, the interacting elements have a behavioural repertoire that usually falls somewhere between that of the basic building blocks researchers often have in mind for artificial chemistry and the complexity associated with multicellular organisms. That is, the elements are usually designed to replicate a few basic properties typical of single cells. These include the ability to:

- deposit chemicals into their environment,
- detect chemicals in their environment,
- move or rotate under their own power,
- adhere to and collide with one another,
- divide / reproduce,
- die
- change their colour, size or appearance.

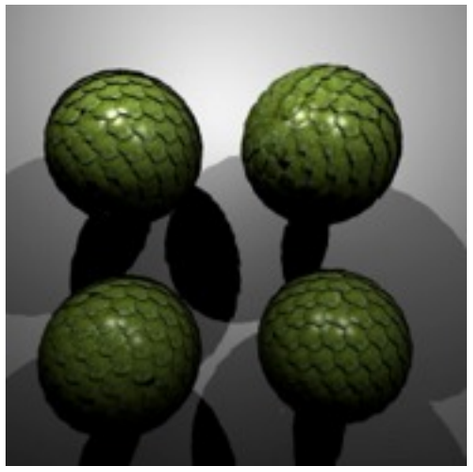
With these few properties, simulated cells can fulfill a range of roles. This section provides an overview of two techniques that simulate cells and their high-level behaviour: one for texturing a surface, and the other for defining a dynamic geometric model.

Cellular textures

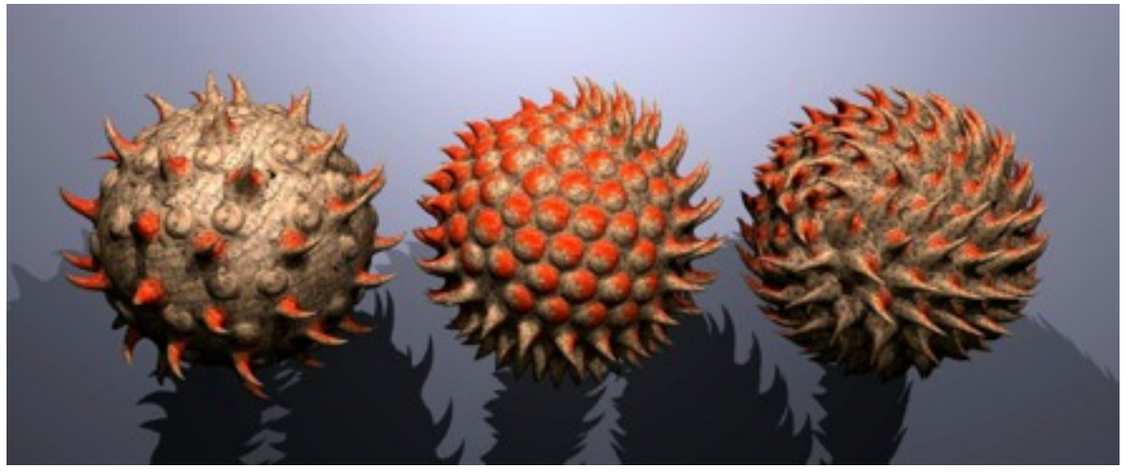
The surfaces of organisms are covered with a semi-permeable membrane. In the case of multicellular organisms this might be an exoskeleton, shell, skin, bark or other protective enclosing layer. The layer might be the substrate on which hairs are mounted, or thick fur, scales, thorns, even feathers. By modelling a collection of cells on a surface it is possible to produce a *cellular texture*; so named by Kurt Fleischer in 1995. Among other applications, Fleischer applied his technique to generate, disperse and orient 3D macro-structures (e.g. thorns, scales and fur) on the surface of larger 3D geometry.

The cellular texture technique takes some of its inspiration from particle systems since cells move, particle-like, across a surface or through space. The technique adopts ideas from reaction-diffusion systems also, since the continuous medium through which the cells move may support the reaction and diffusion of chemicals.

Cells operate under their own cell programs. These are sets of instructions that govern an individual cell's response to its local environmental conditions, including the chemical concentrations at its position, and to the values of its state variables. In response to its program, a cell may perform any of the actions listed in the previous section. In this way it is possible to, for example, have cells that move to a surface, die if they are too far from one, align themselves with a vector field or with their neighbours, find and adhere to other cells, or divide until they completely cover a surface. Cells might also adjust their size or appearance based on some aspect of their environment such as chemical concentrations or geometrical properties of the model on which they are situated.



Scales. The image shows four views of a sphere uniformly covered in cells that have been rendered as scales. The cells divided to cover the surface and died if they were pushed too far off of it. The cells align themselves with their neighbours. © Kurt Fleischer 1995. Used with permission.



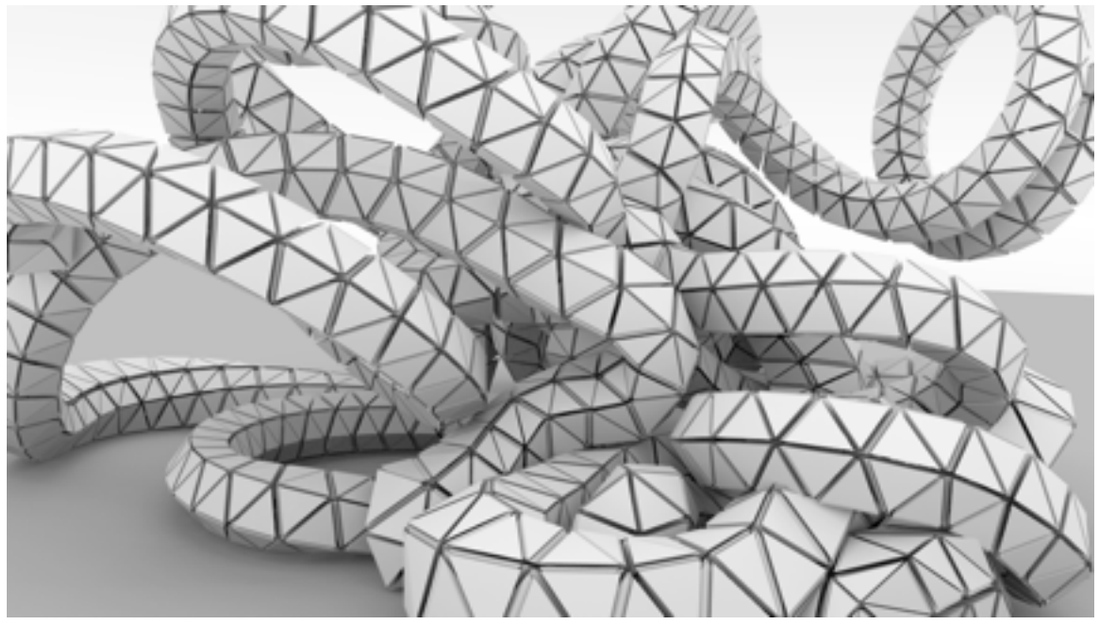
Varying thorns. This image shows three spheres covered in cells that have been rendered as thorns. The thorn geometry is driven by a reaction-diffusion system set up by chemicals that are transferred between cells across contact boundaries. © Kurt Fleischer 1995. Used with permission.

With these techniques Fleischer generated several natural-looking macro-structural textures including scales, thorns and the fur of a bear. These cellular textures smoothly cover the

surfaces of even complex geometric models because they are generated on the geometric surface directly. This is in contrast to standard texture mapping techniques which attempt to cover a surface with an image that is treated like patterned wrapping paper. The standard technique often introduces undesirable creases and discontinuities in the surface pattern. Cellular texturing does not suffer from this problem.

Cellular modelling

Since biological organisms all consist of one or more cells, it is natural to wonder if a model of 3D cellular development might be constructed to mimic the growth patterns of real biology to produce virtual cellular life. Several projects of this type exist under an overarching banner we might think of as cellular modelling. There are many difficulties to be overcome in simulating this kind of development. These include problems whose solutions are relatively well understood but nevertheless remain practically complex or computationally expensive to simulate – such as collision detection and response, friction and other physical interactions between 3D cells. Other problems highlight the current limits to our understanding of biology. For instance, how does an organism, using only chemical triggering mechanisms, ensure that the right kinds of cell appear at the right location at the right time during development? Inroads have been made into solving these problems. In fact, Artificial Life simulations can play a role in assisting us to learn about these aspects of biology.

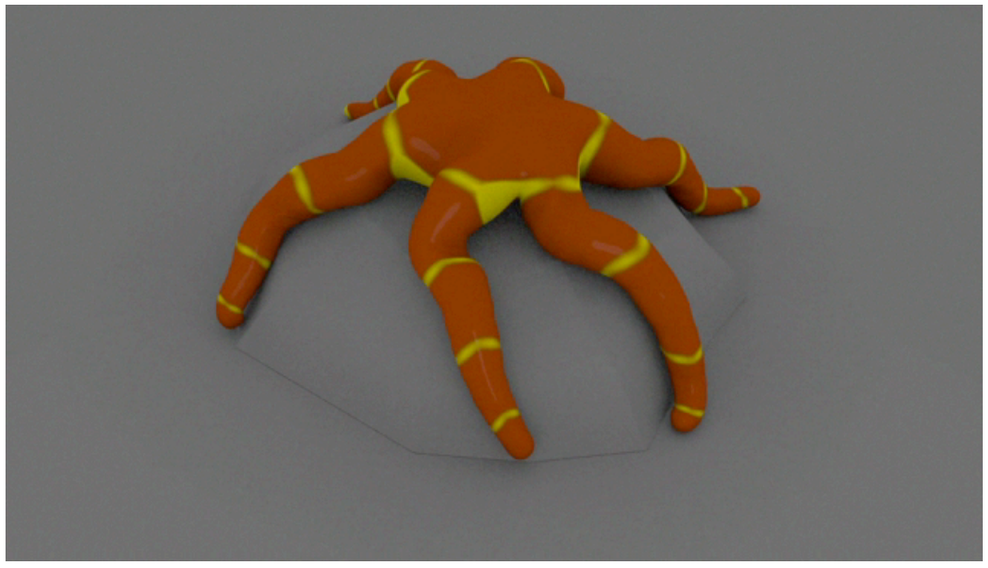


Curling Gravity Tetrahedra. This image was produced using the Simplicial Developmental System (SDS), a cellular modelling technique. © Ben Porter 2010. Used with permission.

One technique that has been applied to simple 2D and 3D cellular-level developmental modelling is L-Systems. It is beyond the scope of this book to examine how this technique is applied in this context, but we will provide an overview of a more recently devised method, the *Simplicial Developmental System* (SDS) by Benjamin Porter.

Organisms are modelled in SDS as a collection of cells represented by point particles. As we noted was typical for cellular systems, the SDS cells may contain chemicals of different concentrations that can be produced or destroyed within a cell. Also as before, these chemicals can diffuse between neighbouring cells across shared edges. Each SDS cell has a program with instructions such as “if my radius is > 2 then divide” or “if the concentration of chemical A is less than 0.5 then pro-

duce chemical B". Where the SDS cells differ from those encountered earlier in this section is in their physical relationships with one another. Each SDS cell is a point mass connected to other cells via edges and a set of springs arranged to form a matrix of tetrahedra – a 3D mass-spring system. A spring connects a group of four particles to form a tetrahedron. A group's springs work to preserve the volume of the shape they enclose in the same way a normal spring tries to preserve its length.



A starfish grown using the Simplicial Developmental System (SDS) is draped over a rock. © Ben Porter 2010. Used with permission.

Further reading

K. W. Fleischer, D. H. Laidlaw, B. L. Currin, A. H. Barr (1995), "Cellular Texture Generation" in *Proceedings SIGGRAPH '95 Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, ACM New York, NY, USA, pp. 239-248.

Porter, B. (2009), "A Developmental System for Organic Form Synthesis", Artificial Life: Borrowing from Biology, LNCS 5865, Springer-Verlag, pp. 136-148.

Chapter 4

Organism Growth

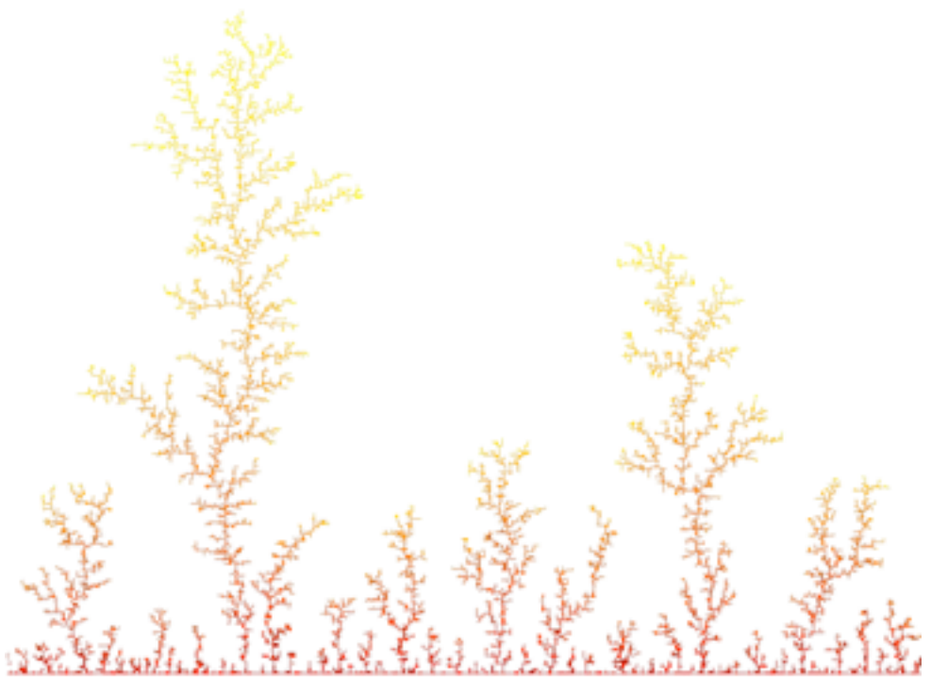
Nature creates complex structures in many ways, but arguably the most impressive is the development of an organism. In this chapter we explore a few models of the growth and development of plants.

SECTION 1

Diffusion-limited growth

TOPICS

1. Diffusion-limited accretive growth
2. Diffusion-limited aggregation

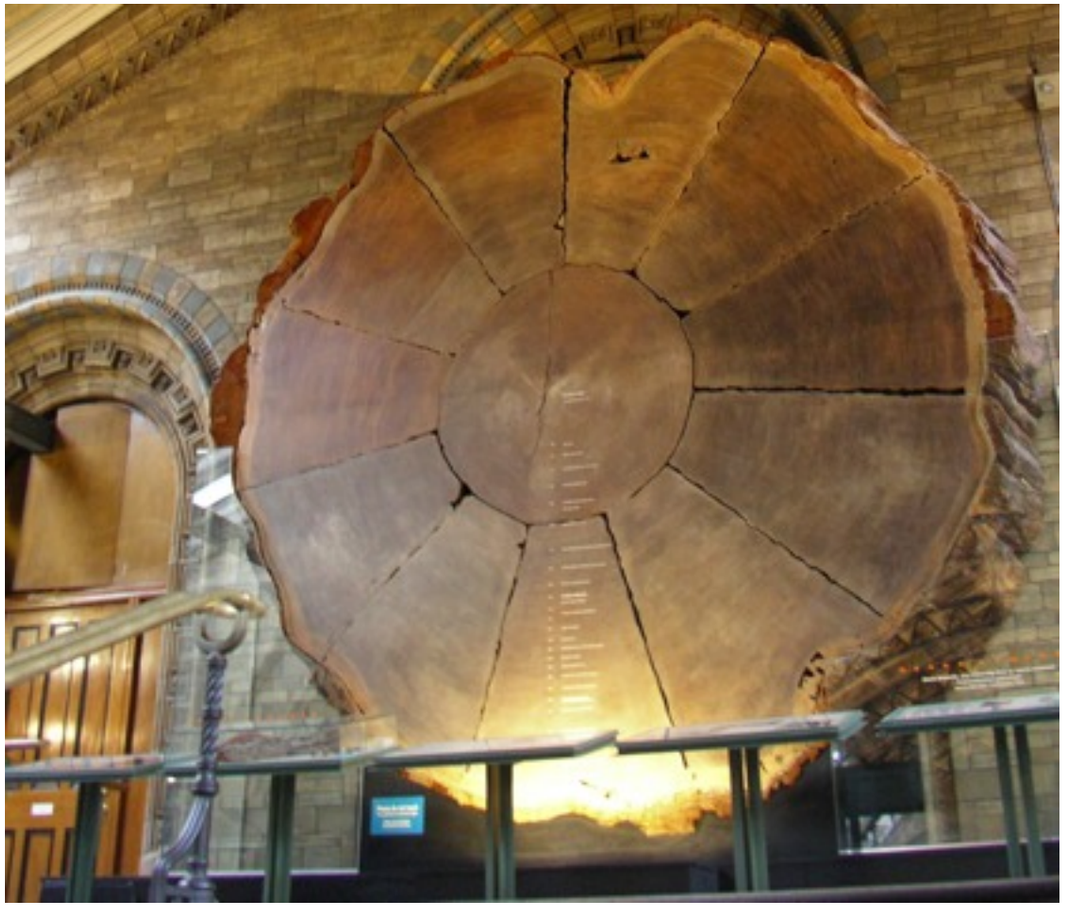


Diffusion Limited Aggregation from a linear seed at the base; simulation result from an applet by Joakim Linde.

Perhaps the most familiar form of growth is the addition of new material around the boundary of an existing core, cell by cell, layer by layer. This kind of process produces the growth rings of a tree trunk over many years. But, perhaps surprisingly, there are processes that are similarly accretive, and yet they generate intricate *branching* structures rather than uniform rings. Diffusion-limited accretive growth is one of these.

Trees have a layer of tissue between the bark and the wood called the *cambium*. This produces the woody cells that make up the timber on the inside of the tree, and the bark on the outside. The production of cells by the cambium is rapid in spring when large cells are formed, slows down in summer and autumn when small cells are formed, and stops during winter. This variation produces the annual growth rings we recognise in the cross-section of a felled tree.

We can understand diffusion-limited accretive growth by considering a single bacterial cell in an **agar**-coated petri-dish. In the vicinity of the cell, the food supply will be depleted, setting up a gradient in the level of nutrition with its low point at the interface between the cell and the agar. Distant from the cell, where the nutrient has been untouched, the nutrient concentration remains at its maximum. If the cell was to divide now, creating a daughter cell fastened to the parent's boundary, the nutrient gradient would be altered as the child and parent continued to eat. The lowest point of the gradient would now be near the junction between the two cells, since each cell is consuming nutrient from that location. As each of these cells now divides when it has acquired sufficient resources to do so, the



A slice from a 1300 year-old giant Sequoia, from the Sierra Nevada in California, USA on display in the Museum of Natural History, London. The centre-most of the tree's growth rings were laid down in the year 557 CE. The tree was felled late in the 19th century.

nutritional gradient continues to change. The only location for new daughter cells is at the existing structure's boundaries, this is the only place where the existing cells can acquire sufficient space and nutrients to reproduce. Near the oldest cells on the plate the space and nutrient supply have been depleted.

The pattern that results from this process is not a series of rings, it is a branching structure. This appears because cells on the outskirts do better by staying away from neighbours than they do by overcrowding and trying to eek out a living

Diffusion Limited Aggregation from a central seed; simulation result from an applet by Joakim Linde.

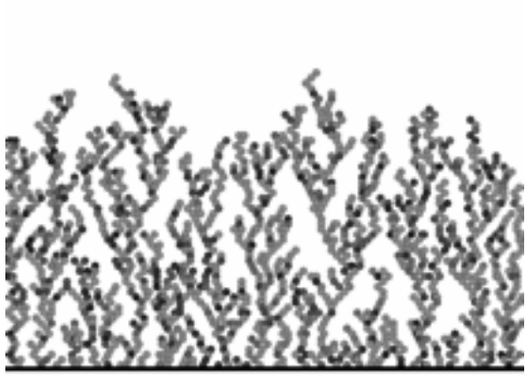
where the nutrient supply has been exhausted.

A discrete interpretation of diffusion-limited accretive growth called *diffusion-limited aggregation* (DLA) is easy to simulate algorithmically.

[**Try the interactive widget online**]

The process of aggregation can begin with a single fixed seed in the center of an empty space, or with a row of fixed seeds organised along a boundary. Free-floating particles are then introduced to the empty space. These execute a "random walk", hopping and jumping about until, at some stage, they bump into the side of one of the fixed seeds. When they do this, the free particles are permanently fixed to the seeds at the location of their contact. They now form part of the fixed structure to which new free-floating particles adhere. As more randomly moving particles stick to the fixed structure, it grows

A DLA-like pattern is found in dendrites. These pseudo-fossils are (inorganic) oxides that crystallise in branching structures.



A snapshot of a DLA simulation.

branches. Clearly it becomes more difficult for the structure to grow near its center since the most likely place for randomly moving particles to first contact the fixed cells is at the structure's margins. A structure begun from a central fixed seed is provided in the illustration at left, one commencing from a line appears at the start of this section.

The DLA algorithm simulates the deposition of ions on an electrode. It is not an accurate model of the development of organisms such as bacteria on an agar plate, as the biological organism is responsible for actively consuming the nutrient supply in its environment. The result is still an interesting growing **fractal** all the same.

Further reading

Prusinkiewicz, P. (1993). "Visual Models of Morphogenesis." *Artificial Life* 1(1-2), pp. 61-74.

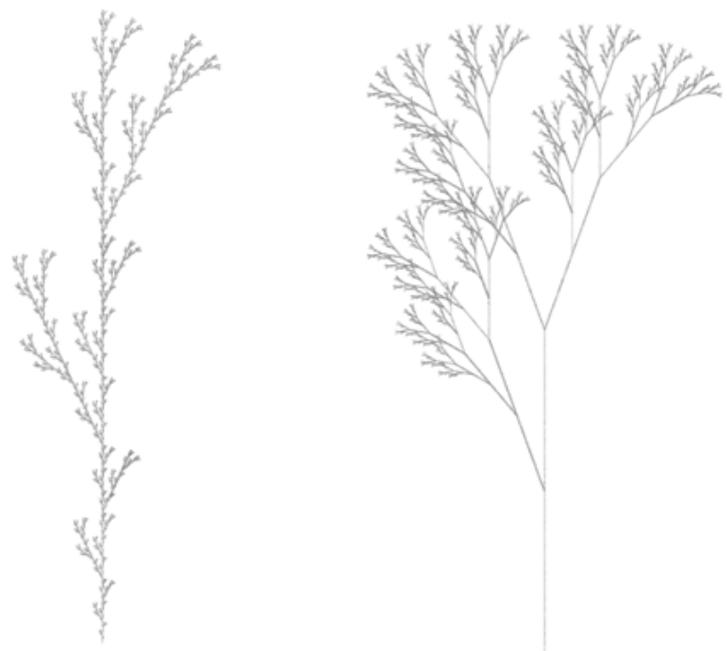
Witten, T. A. and L. M. Sander (1981). "Diffusion-Limited Aggregation, a Kinetic Critical Phenomenon." *Physical Review Letters* 47(19), pp. 1400-1403.

SECTION 2

Lindenmayer systems

TOPICS

1. What are L-systems?
2. Turtle graphics
3. Simple extensions to L-systems



L-Systems are capable of generating many subtle tree-like structures. The 1990 text by Lindenmayer and Prusinkiewicz, *The Algorithmic Beauty of Plants*, is a classic not only in Artificial Life but in Computer Graphics modelling. It is available free online.

Plants are photo-synthesising *autotrophs*; that is, they use energy from the sun to generate their own energy for development and reproduction. The energy is captured by the green chlorophyll within a plant's leaves, but only when the leaves are exposed to sunlight. Hence a plant requires some kind of semi-rigid structure to allow it to position its leaves in the sunlight: a trunk and branches, a vine or stem.

What are L-systems?
[Try the interactive widget online]

A *Lindenmayer system* (L-system) is a method suitable for generating plausible plant branching structures using simple rules. The technique is named after Aristid Lindenmayer (1925–1989), the Hungarian biologist who came up with the idea in 1968.

To build a model plant from an L-system we generate a string that represents the branching structure, and then interpret this string graphically as a collection of sticks, stems, leaves, flowers or fruit. The string is generated from an initial seed by a set of production rules that describe how to replace its characters with new ones in discrete steps. The system is therefore based on a *grammar* for generating strings that represent plant structures. Depending on which set of rules is used, and on the seed from which an L-system is started, it is possible to generate many different branching structures of great complexity.

Here, as an example, are some L-system components.

Alphabet: a set of symbols representing the *modules* of a plant.

A = B =  C = 

Axiom: the initial string representing the "seed" of the L-system. The axiom consists of one or more symbols from the alphabet.

A

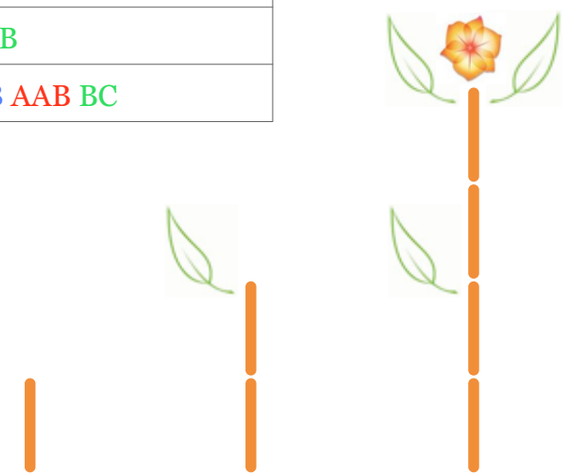
Production rules: a set of rules that replace a symbol in the string (called the *predecessor* module) with zero, one or several other symbols (called *successor* modules).

	A → AAB	
predecessors	B → BC	*successors*
	C → C	

The L-system generates a string in discrete steps. At each step, the appropriate production rule is called upon to replace each character in the string, beginning with the axiom at the first step and proceeding until a sufficiently long string has been generated to produce a plant model of the desired complexity. Different alphabets, production rules and axiom can be used to generate different plant models. For the alphabet, axiom and rules just given, here is a sample expansion over 3 steps. The second stage of the replacement is colour coded to highlight the conversion of each character in the string by a single production rule. One possible graphical interpretation of the sequence is given below.

There are many possible graphical interpretations for these

Replacement step	String
0	A
1	A A B
2	AAB AAB BC



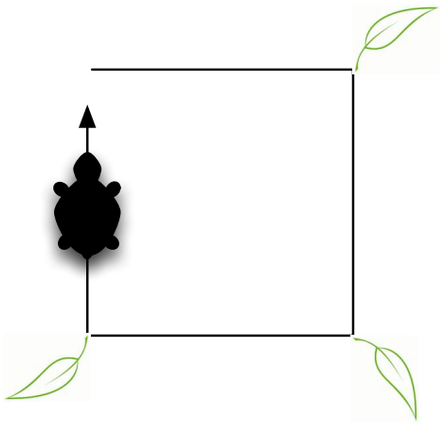
strings. Nothing about the strings themselves indicates where the modules their characters represent must be drawn on a page or with respect to one another. Nevertheless, some conventions for interpreting strings as useful plant structures are often observed. One of these is based on a "turtle".

Turtle graphics

Imagine a (virtual) turtle roaming a page or screen. The turtle may be instructed to raise or lower its pen, to move ahead a distance, to turn by some amount, to save its position, orientation and pen conditions by pushing them onto a **stack**, and to

pop this information back off the stack when requested, returning the turtle to a previously stored state. Collections of turtle instructions can be stored as subroutines. These allow the turtle to store detail for drawing complex structures that will be repeated. For instance, a leaf or flower might need to be drawn multiple times so it makes sense to have a subroutine for each.

> The *Turtle* can be traced back to the work of Seymour Papert, a maths professor interested in educational tools for children. In the late 1970s, he proposed that children would enjoy and benefit from learning to program a computer in the LOGO programming language, by having their programs executed by a Turtle like that described here.



```
Set colour to black
Pen down
Do 4 times
{
   Forward 10 units
   Push state
   Set colour to green
   DrawLeaf ( )
   Pop state
   Right turn 90 degrees
}
```

Instructions for a turtle that generate a square with leaves at each corner by calling the pre-existing subroutine DrawLeaf ( ).

It is convenient to use turtle graphics to interpret L-system characters graphically. Each character in the string is treated as a turtle instruction. For instance, **A** might tell the turtle to draw a stick aligned with its current direction. **B** could cause the turtle to execute a subroutine to draw a leaf, and **C** might instruct the turtle to draw a flower at its current location.

To allow the L-system with its turtle interpretation to produce branching structures we can introduce brackets **[** and **]** to the alphabet. These cause the turtle to push and pop its current state onto the stack respectively. By following **[** with an instruction to turn the turtle left (+), or right (-), we can make branches in either direction, and then return to continue drawing from where we left off with a **]**. Here is a simple example.

Axiom: A
Production rule: A → A [+A] A [-A] A

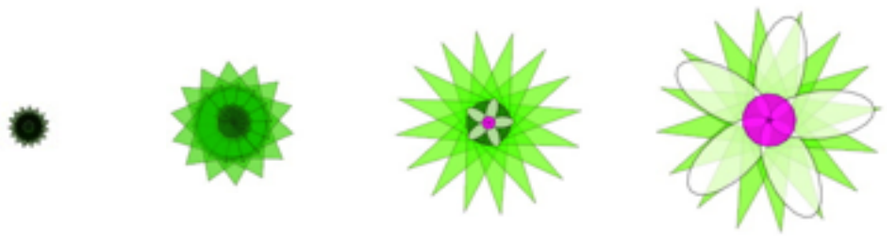
At left is the result of having the turtle draw the string that results from applying this rule 5 times. The turtle was rotated 25 degrees for each turn instruction.

Simple extensions to L-systems

Sometimes we might wish for a plant structure to grow smoothly over time instead of in discrete jumps corresponding to the sudden appearance of a new character in the string. One approach to achieve this is to add a parameter to a symbol in the alphabet. This can be represented as a subscript that is explicitly incremented within a production rule.

$$X_i \rightarrow X_{i+1} \quad \text{where} \quad X \in \{ A, B, C \} \text{ and } i \geq 0$$

The character with the subscript might then be drawn as a gradually lengthening leaf or as stages in the opening of a bud. The parameter's value specifies the developmental stage of the represented module. Unsurprisingly, these are called *Parametric L-systems*.



F_0 F_1 F_2 F_3

In each example above, exactly one production rule is provided for every alphabet symbol – these are *deterministic L-systems*. Non-deterministic L-systems are also useful. In these, there may be several production rules available to act on a particular symbol. In cases where there is ambiguity, the choice of rule can be determined in a number of ways:

A delay mechanism: ambiguity is avoided in the selection of production rules by adding a mechanism to the grammar which, after some number of iterations or applications of a rule, replaces a production rule by another. This might allow a plant to switch from a state where it is producing leaves, to a flowering state.

A stochastic mechanism: ambiguity is avoided by specifying a probability with which each ambiguous rule is chosen to operate. This allows variation between and within plant models – useful if you want to generate a forest of different trees!

Environmental change: an entire set of production rules may be altered to another set after some external factor triggers the change. For example, a tree might make the switch from leaf production to flower bud production when an environmental model enters Spring.

Further reading

Prusinkiewicz, P. and A. Lindenmayer (1990). "The Algorithmic Beauty of Plants". New York, Springer-Verlag (free online).

An L-System generated tree structure.

SECTION 3

Voxel automata

TOPICS

1. Environmentally-influenced growth models
2. Overview of voxel automata
3. Voxel automata growth rules

An image generated using voxel automata grown over a model of a gazebo. © Ned Greene, 1989. Used with permission.

A strangler fig that has strangled a now deceased tree, Queensland, Australia.

The propped up branches of the the Tree of Hippocrates. Kos, Greece.

The study of real plants highlights the extent to which they are subject to their environment. They might be blown into unusual shapes by the prevailing winds. They can grow over

A tree and vines grow from the roof and walls of a church. Melbourne, Australia.

Snow Gum, Snowy Mountains, Australia. © Richard Thomas 1995. Used with permission.

rocks and support themselves on walls. Vines can swallow architecture, and trees in their entirety. These aspects of a plant's form can be among the most dramatic of a landscape's traits. Unfortunately, many methods for modelling plants do not easily allow any real interaction between a developing plant model and its environment. This can be a serious drawback.

Overview of voxel automata

Ned Greene's *Voxel automata* technique (originally called *voxel-space automata)* closely marries plant development and an environmental model, allowing for some unique plant structures to be grown procedurally. In particular, it has been used to generate complex vines that carpet architecture and respond appropriately to virtual sunlight.

Voxel automata naturally account for:

- Phototropic effects that determine the rate and direction of growth.
- Growth around obstacles such as rocks, walls, paths and other plants.
- Self-intersection avoidance of plant structures.

Vines generated using voxel automata. © Tobias Gleissenberger, 2006. Used with permission.

Voxel automata simulations are initiated by dicing the virtual space into small cubes called **voxels**. Voxels that contain obstacles to a plant's growth such as rocks, architecture or pre-existing trees are identified. These are marked as "filled" in a process called *tiling*. Filled voxels will not be available for a plant model to grow into.

Plant seeds are next placed around the environment in as many empty voxels as desired. These are the initial active nodes of plant development. Now, at discrete simulation time steps, a potential direction and distance of growth is determined by applying growth rules at each active node. A short plant segment is then laid into the voxel space in a suitable direction and, possibly twisted with respect to its parent.

Lastly, voxels newly occupied by plant segments are marked as "filled" to indicate that future plant segments may not grow into them. This simulation proceeds a step at a time, laying down plant segment after segment in accordance with the rules until a sufficient amount of vegetation has been generated.

Winter, generated using voxel automata grown over a model of a gazebo. © Ned Greene, 1989. Used with permission.

Growth rules

How does each active node of a plant model determine the direction in which to grow next or whether or not to branch or grow a leaf?

Each plant has a set of parameters (see sample Table below) that determine the characteristics of its skeleton, leaves, and the lighting or environmental conditions it prefers. The simulation randomly generates many, possibly over a hundred, potential segments from each active node. It then selects from these the suggested segment that best fits the preferred conditions parameterised for the model plant. A proposal for a plant segment is accepted if it satisfies tests for collisions against existing objects in the voxel array, proximity to other elements of the structure or environment, and the amount of sunlight falling into the cells it occupies.

Outline of a plant skeleton's parameters

segment length	The length of each plant segment.
branch age range	The average number of growth steps between new branches.
branch angle	The angle of each new branch.
vertical bias	The tendency of the plant to grow upwards.
maximum no. of trials	The maximum number of trials to place a plant segment before giving up.
minimum no. of trials	The minimum number of trials to place a segment before selecting one with the best proximity to a target.
seek proximity	The preferred proximity (in voxels) to a tiled voxel.
max. proximity	The limit beyond which a new segment is considered too distant from a tiled voxel to be viable.

Values for light exposure of a plant segment are computed in voxel space by estimating the amount of direct sunlight entering a voxel into which growth is suggested; and by estimating the amount of sky-dome exposure a voxel receives. Shadows caused by tiled voxels between the arc swept out by the Sun and the voxels to be occupied by the proposed plant segment are taken into account. Shadows caused by tiled voxels between the proposed growth voxels and the sky dome are also. In this way an estimate of direct and ambient light at a location can be approximated.

Heliotropism may be simulated by incorporating a Sun-seeking bias into the rule set. A plant may also be made to grow over or around an existing geometric model tiled in voxel space. Regions of the model may alter the growth characteristics of the plant. For example, paving stones may forbid growth over their surface, gables may encourage growth along their length or arches may encourage vines to crawl around them.

Further reading

Greene, N. (1989). "Voxel Space Automata: Modeling with Stochastic Growth Processes in Voxel Space", in Proceedings of SIGGRAPH '89, pp. 175-184.

SECTION 4

Particle systems

TOPICS

1. What are particle systems?
2. Particle system components
3. Particle system simulation
4. Rendering

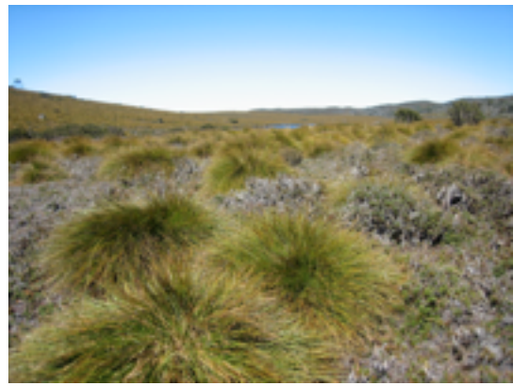
Grass in a landscaped garden, Shanghai, China.

Cabbage trees, Tasmania, Australia.

What are particle systems?

When we examine a natural scene, say a landscape, from a distance, much of what we perceive consists of large shapes and solid structures. These can often be represented geometrically as solid objects with well-defined surfaces. But in fact every-

Wind-blown grass.



Button Grass tussocks, Tasmania, Australia.

thing we see is made of tiny particles. Sometimes this is true only at the level of molecules, atoms or sub-atomic particles. In cases like this, the molecular bonds construct a well-defined solid surface. But many natural phenomena are particulate at a much higher level, one that does not involve stiff bonds *between* particles.

Weeping Willow tree, Melbourne, Australia

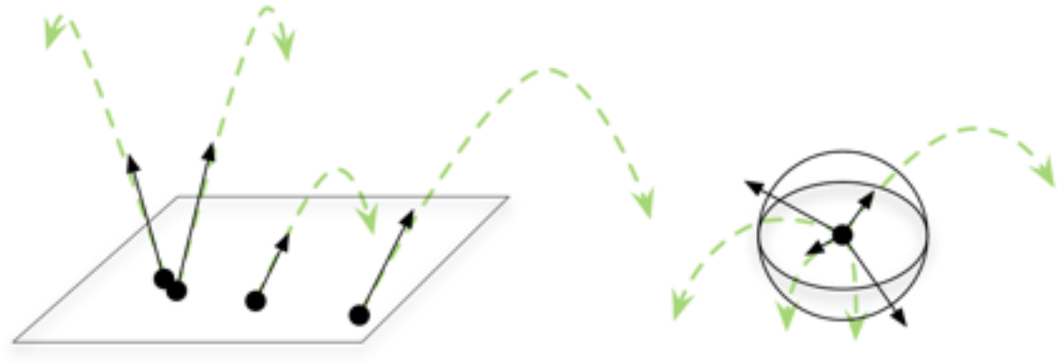
That is, although we might observe a single phenomenon or entity such as a cloud, it is actually made up of countless macro-sized particles moving independently, often under the influence of an external force such as gravity or the wind. Obvious examples of these *particle systems* include fog, smoke, clouds and sand dunes. A shower of sparks from an exploding log, a metal grinder or a firework is also particulate. In these cases it makes sense to base our computer models of the phenomena on particles. This idea was introduced to computer graphics in 1983 by William Reeves. He used the technique to model a wall of fire travelling across the surface of a planet, fireworks, and grass. Here we will focus on modelling blades of grass but the technique is equally applicable to stems, willow fronds or other long, flexible plant structures. The method requires computing the trajectory of many particles ejected into space from a *generation shape*. After emission, each particle falls under the influence of gravity. The particles' trajectories sweep out paths along which plant structures can be generated. In a later section of the book dedicated to physical simulation we will look in more detail at the steps involved in this process. Here we treat the particles' movement in the simplest way we can to achieve the desired effect.

Particle system components

We only need to consider simple particles for this application. Our particles will need:

- size, shape and colour
- position and velocity vectors
- birth time and life span

We also need a particle *generation shape*. This is geometry from which particles will be ejected. To make a field of grass, a rectangle lying flat on the ground would be an appropriate shape from which to launch particles. To produce leaves or fronds drooping in a clump from a branch tip, perhaps a single point would be an effective particle source.



Rectangular (polygonal) and spherical (point) particle generation shapes. Black arrows indicate initial emission velocities of particles. Green arrows indicate paths swept out by particles over time under the influence of gravity. These green paths can be used as the spines of blades of grass, leaves or fronds.

A generation shape will need:

- A location in space and possibly an extent specified as a line, polygon or volume.
- A rate at which to generate new particles.
- A start time at which to begin creating particles.
- An end time at which to stop creating particles.

Particle system simulation

A simulation to generate paths from particles proceeds in discrete time steps. At each one of these steps three things happen:

(A) Active particles that have reached the end of their life span are "frozen" (they won't be updated further);

(B) New active particles are introduced from active generation shapes;

(C) Active particles are moved through space according to a physics simulation.

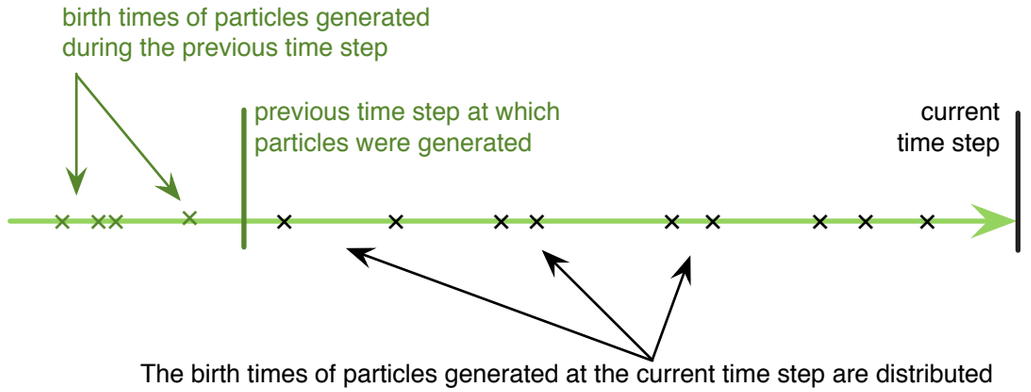
This process is repeated many times until there are no active particles and no generation shapes waiting to generate new ones. The example detailed below discusses steps A, B and C in the context of generating blades of grass, but the same principles apply to other models.

A. Old particles are frozen.

The collection of existing particles in the simulation must be tested to see whether it has reached the end of its life. This allows us to decide which particles need to be frozen in space. In the case of generating a field of grass, the end of a particle's life tells us to stop generating its path. The path that the particle has swept out so far is maintained.

B. New particles are introduced.

1. Each particle begins life at, on or in a generation shape. In the case of a rectangular generation shape a position is randomly generated for each new particle somewhere within the region. The number of particles generated from an active generation shape is determined by a rate parameter, taking into account the time elapsed since the previous simulation time step.
2. Each particle will have its birth time set to a random time between the previous time at which particles were generated by the generation shape, and the current simulation time. This ensures that particles are not generated in bursts but have evenly distributed birth times.

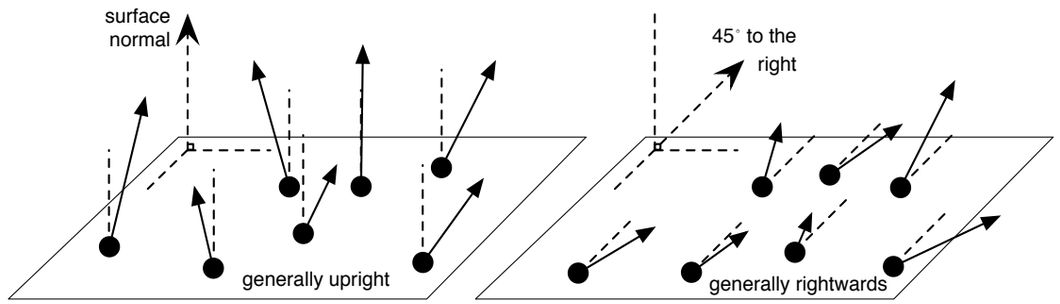


The birth times of particles generated at the current time step are distributed across the time elapsed since the last batch of particles was generated

If particle birth times aren't properly distributed over a time step, particles will appear in the simulation only on time step boundaries giving the appearance of the generator releasing bursts of particles instead of a steady stream.

3. The lifespan of each new particle is set to an appropriate value to ensure an adequate blade length will be swept out. The values might be varied between particles to generate blades of different length.
4. Each new particle is given an initial velocity. This vector will have a magnitude (i.e. the particle's speed) distributed

randomly about a predetermined mean. The amount of variation alters the variability of the grass blades' heights before they bend downwards under gravity. The velocity vector might also have its direction distributed randomly about a predetermined mean. If we want the grass blades to generally grow upwards, then the surface normal of the rectangular generation shape would suffice as a mean. However if we want the grass to lean in one direction as if it had been blown by the wind, we can use any suitable vector as the mean direction.



Each particle's initial velocity can be set individually. By varying initial speeds and directions the grass may be made more or less uniform.

5. A particle's colour is initialised to the colour of a grass blade's base.
6. Each particle's shape is set to the cross-section of a grass blade's base. Most simply, a line, a **V**, or an **O** shape might suffice. If necessary, the size of this should be set to the desired size of the blade's base.

C. Active particles are updated.

1. Each active particle must have its velocity updated by applying acceleration due to gravity. The change of velocity can be calculated using Euler's formula:

$$v' = v + a\Delta t$$

which simply states that we need to multiply the acceleration due to gravity – a, a constant vector of magnitude 9.8 ms^{-2} oriented downwards – by the simulation time step length (Δt, a scalar value) and add the resulting vector to the velocity vector v of the particle to give us the particle's new velocity v'.

2. Each particle must have its position updated by its velocity. The change of position can also be calculated using Euler's formula:

$$p' = p + v\Delta t$$

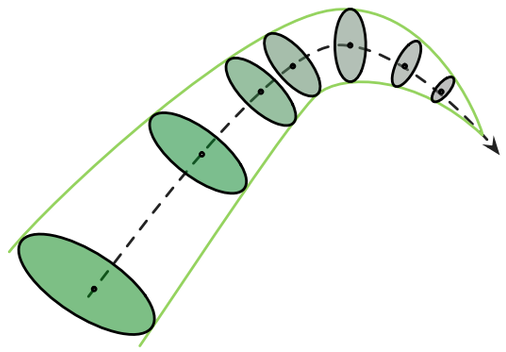
which in this case simply states that we need to multiply the velocity v of the particle by the time step length (Δt, a scalar value) and add the resulting vector to the current position vector p of the particle to give us its new position p'.

Particles having their velocity and position updated using Euler's formula include those just introduced in the current simulation time step (§B.2). These particles will have Δt less than a full simulation time step that is calculated by subtracting their birth time from the current simulation time.

3. We must update the colour of each particle to give us the desired change since the previous time step. For instance, some grasses are white at their base, green in the middle and brown towards their tips. To simulate this, as each particle ages, we must compute its new colour by interpolat-

ing between white and green for the first half of the particle's life, and between green and brown for the second half of its life.

4. We must update the size of the blade by adjusting the size of the particle. For example, some blades are broadest at their base and taper towards the tip. To simulate this we simply reduce the diameter of the particle as it ages so that when it reaches the end of its life the size will be zero.



The diameter of the particle can be reduced over its life time. In this way, the path it sweeps out will taper towards the growing tip.

5. Since the particle's movement is sweeping out a path along which we will lay a blade of grass, we need to append the current position, colour and size of a particle to the list of all previously computed values for that particle. Don't throw anything away or overwrite previously calculated values!

Rendering

We can render the paths of all particles while the simulation is running. This will give a simple animated representation of growing grass. Alternatively, the rendering can be postponed until all of the particles and generation shapes have become inactive. At this point we can stop the simulation.

To render a particle's path we interpolate between each stage of its saved trajectory. This can be done most easily with a line of thickness specified by the particle's size and colour specified by the particle's colour at each stage of its history. If more complex forms are required, polygons or surfaces of appropriate cross-section generated from splines can generated between subsequent steps of each particle's trajectory.

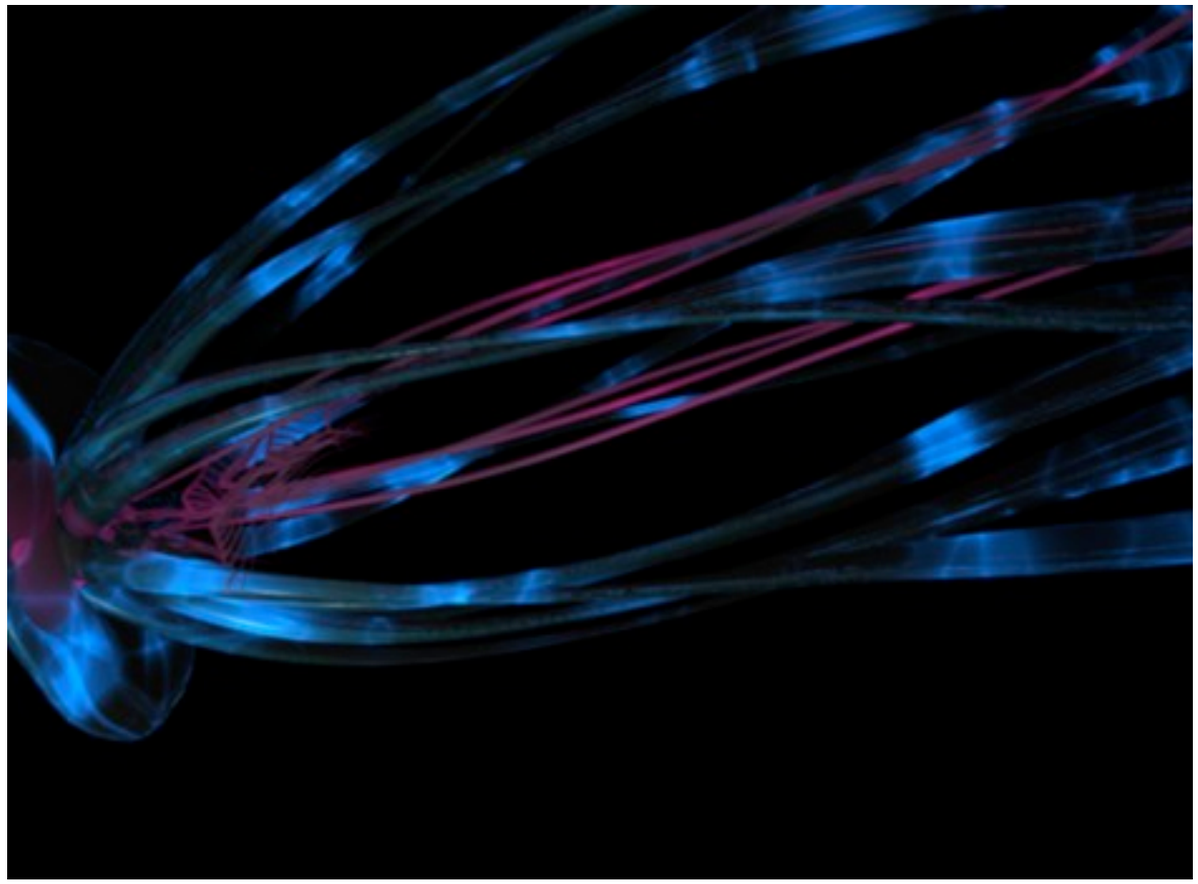
Further reading

Reeves, W.T. (1983). "Particle Systems – A Technique for Modeling a Class of Fuzzy Objects", ACM Transactions on Graphics, Vol. 2, No. 2, pp. 91-108.

CHAPTER 5

Locomotion

Creatures as tiny as a single cell or as large as rhino can move about. They twitch, crawl, row, flip, slither, gallop, charge, walk, soar, hover or hop to name just a few mobility approaches. To understand how these movements are possible, some basic physics is necessary. Although these principles are not specific to Artificial Life, an introduction is included here to allow readers to understand the basics.



SECTION 1

Physical simulation

TOPICS

What is physical simulation?

Since at least the 7th C. BCE, articulated dolls capable of bending a limb have been made. But getting them to move autonomously, and to coordinate each joint to mimic the fluidity of animal motion has always been challenging. This same hurdle must be met by today's robotics researchers as they try to build articulated machines that walk, catch a ball or lift a fragile object without crushing it. Each mobile biological organism has evolved the physiology that allows it to coordinate its motion. We will explore the process of evolution of behaviour in a later chapter. In this one we explain how to simulate the physical processes that allow locomotion and movement to occur in the first place.

Physical simulation is a relevant skill for many software-based artificial life researchers because the unique structures of real organisms are often adaptive traits for survival and reproduction in a physical world. If we want similar diversity in a virtual environment, one way to get it is to simulate real physics.

In the virtual world nothing comes for free, especially not the basic properties of the physical universe. We must therefore explicitly state all constraints: if we don't want solid objects to intersect one another we must explicitly code that; if we want items to fall under gravity we must write software to make it happen; if we want a force to accelerate an object we must understand and implement this process; if we want virtual creatures to swim we must model fluid drag.

Here are four stepping-stones towards physical models of complex vertebrates.

1. Differential equations
2. Particle dynamics
3. Rigid-body dynamics
4. Articulated-body dynamics

The astute reader will notice that particle systems have already been discussed in relation to modelling grasses and leaves. In that discussion, the solution of differential equations was introduced surreptitiously also. Particle systems are in fact simple physical models. The particle system discussed earlier was a *kinematic* model. Kinematics is concerned directly with the acceleration, velocity and position of bodies (such as particles), without reference to their mass or the forces required to move them. In this section we will discuss particle *dynamics*. Dynamics simulations take into account the effects of forces applied to bodies and the resultant changes in acceleration, velocity and position they undergo. As this is not a text on physical simulation we will only go so far as to explain particle dynamics and demonstrate how this may be used to simulate simple moving organisms. Rigid and articulated-body dynamics are beyond the scope of this book.

Differential equations

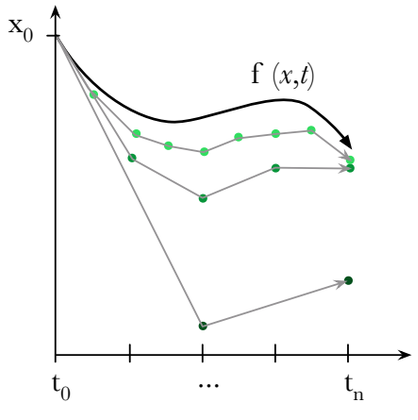
A differential equation is a mathematical way of expressing how some function of a number of variables changes. For instance, a differential equation may specify how a flying particle's position specified as a single Cartesian coordinate, x, along an axis changes over time. Of particular interest to us are *initial value problems* where we are given x at an initial time t_0 and need to follow x over time as we step through a series of time steps from t_0, t_1, t_2 ... t_n where the difference between subsequent times steps, the time step length, is usually shown as Δt. Δt represents the smallest increment of time we consider in a simulation. All events happening within a single Δt of time between time step t_n and t_{n+1} are considered in the simulation to be simultaneous. We can write the initial value problem specifying the position x at time t_0 as:

$$x(t_0) = x_0$$

And the change in x over time is written as:

$$x' = f(x, t)$$

That is, the change in x, specified by x', is a function f of two variables, the position x and the time t.



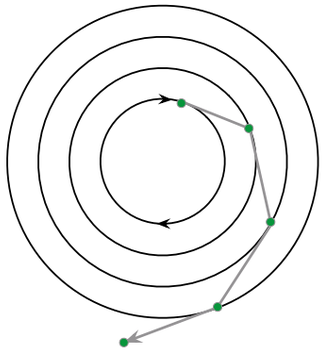
As the function f is approximated by increasingly short line segments, the potential accuracy of the approximation increases.

Euler's integration method

A simple, rough, but often useful way to solve initial value problems specified as ordinary differential equations is to apply a technique devised by the 18th C. Swiss mathematician, Leonhard Euler. Euler's method for approximating a smoothly changing variable

(such as position of a particle x) is to approximate the function by a series of discrete steps, each of which is linear. The smaller the step size (Δt in our terminology), the closer the linear approximation sits to the actual value the function sweeps out. Of course the trade off in using short straight segments to approximate a curve is that you need many of them, especially if the curve changes direction often. This can require great computational resources to manage.

The *Runge-Kutta* integration technique is almost as simple to implement as Euler's approach but it is more accurate – a fourth order approximation. It is also relatively fast to execute. For situations where Euler integration is proving troublesome, Runge-Kutta integration is often suitable. It is important to realise though that even the Runge-Kutta technique is an approximation. Consult a textbook on numerical methods for details.



In some cases, such as when the function to be approximated is a circular vortex in which the direction a particle should move is perfectly circular, it doesn't matter how short the line segments become. The particle will always spiral outwards from the centre.

Regardless of the length of line segments used, if the function being approximated is not actually made of linear segments, then even short linear approximations to its value can never follow its curve exactly. (This is especially problematic when a path is circular. No matter how short the segments you use to approximate a circumference, your approximation will get worse and worse over time unless you correct for your errors.) But the approximation given by Euler's method is often sufficiently accurate for Artificial Life creature simulations that are only to be evaluated visually. Just don't use Euler's method for critical simulations of real world situations though.

For our problem, Euler's method is specified:

$$x(t+\Delta t) = x(t) + f(x,t)\Delta t$$

This says that the value of x at some future time (the current time t plus Δt), is equal to the sum of the value of x at the current time t, and the *derivative* of x multiplied by the time step length. The derivative of x is the *rate of change* of x. The rate of change of position (the value we are interested in for this example) is of course, velocity. So the future position x is given by the current position plus the velocity multiplied by the amount of time over which it acts. This is just what we did in the particle system simulation earlier when we used the formula:

$$p' = p + v\Delta t$$

We were integrating the velocity to compute the change in position. Note that during Δt the velocity is assumed to remain constant – we only use one value for velocity in each time step. Hence, the position of the particle changes *linearly* during each time step. This linearity is the reason why Euler's approximation is called a *first order approximation* of the function. As we indicated earlier, Euler's method represents the change of x as a series of linear increments.

As we know from our particle system discussion earlier, the rate of change of velocity, that is the derivative for the velocity, is the body's *acceleration*. We can use Euler's technique to compute the new velocity of a body at a future time based on its current velocity and the length of time over which the acceleration acts. As before, by using Euler's method we are assuming that acceleration is constant during a time step Δt. We write the change in velocity as:

$$v(t+\Delta t) = v\big(t\big) + f'\big(x,t\big)\Delta t$$

The *f ′* in *f ′(x,t)* represents the derivative of the velocity, the acceleration. So far this is exactly as it was for the particle system simulation. When modelling a falling particle the force acting on it due to gravity *is* constant over time. But what if forces acting on the particle change?

Newton's second law of motion

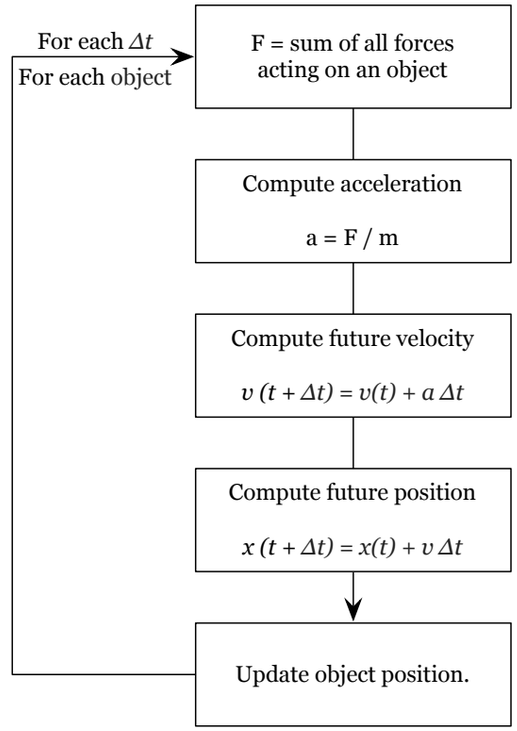
In 1687, the physicist Isaac Newton published his second law of motion. This stated that the acceleration of a body is directly proportional to, and in the same direction as, the net force acting on the body, and inversely proportional to its mass. This is encapsulated in the well known formula:

$$F = ma$$

where F is the net force acting on the object, m is the object's mass and a its acceleration.

What kinds of forces might act on a body? Some examples include forces generated by thrust from a jet, a push or a pull from a spring or muscle. Repulsion and attraction acting between magnets and charged particles are also forces. One moving body might give another a shove, and gravity acts on bodies to bring them together.
Friction between surfaces, including fluid drag between a solid and a liquid, might also be forces considered in dynamics simulation.

The dynamics simulation loop and rendering



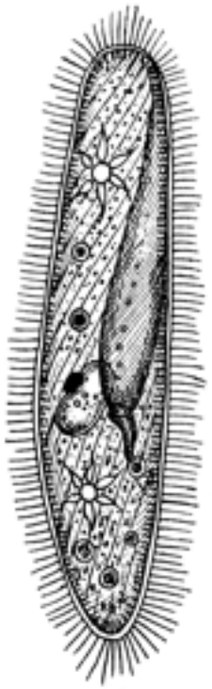
We now have all the pieces needed to implement a simple particle dynamics simulation. For each mobile particle, for each time step in our simulation, we must follow the sequence of calculations just outlined. These are summarised (using Euler's integration technique) in the figure. Each step allows us to plug the computed value on its lefthand side into the righthand side of the next step to continue the calculation.

Recall that when we used particle kinematics to sweep out the span of a blade of grass we drew every stage of a particle's path connected to its previous position? If we are only interested in the current position of each particle we needn't do this. Instead, after a number of simulation time steps we may

just clear the display area completely and re-render each particle in its latest position.

Micro-organism locomotion

The dynamics governing the movement of micro-organisms is quite different to that of familiar marine, terrestrial or avian creatures. Although it is beyond the scope of this book to detail the mathematics, a general explanation of the differences between the kinds of locomotory systems usually encountered, and those that occur at the micro-scale is included next. Assumptions based on our own personal experience do not necessarily hold for other organisms.

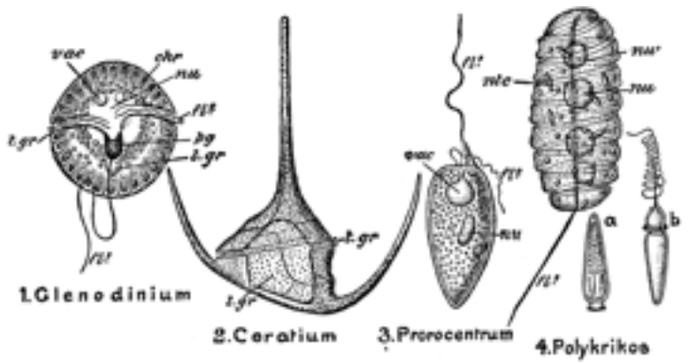


Paramecium is a freshwater ciliate whose surface is covered in tiny cilia.

From Thomson J.A., *Outlines of Zoology*, New York, NY: D. Appleton & Company, 1916.

Protozoa have extremely low masses and must propel themselves through fluids under the influence of high viscous drag. Under such conditions, a microscopic organism cannot successfully propel itself like a cephalopod (octopus or squid), nor can it generate lift with aerofoils in the manner of larger marine or avian animals. Conventional rowing motion is also ruled out at this level. In order to understand how a micro-organism generates thrust, we must examine the behaviour of a fluid/body system at the cellular scale.

The Reynolds number, *Re*, is a dimensionless value representing the ratio of inertial to viscous stresses in a fluid/body system. Systems consisting of a Protozoan and a fluid with the density and viscosity of water have extremely small Reynolds numbers. The interactions between the fluid and solid body are governed almost exclusively by viscous forces. Consequently a single-celled organism in water will come to a halt as soon as it ceases to apply thrust – imagine yourself swimming through honey. Also at low Reynolds numbers, a slow stroke through the fluid induces the same amount of drag as a rapid stroke. If a microorganism tried to “swim” by executing a recovery stroke that is symmetrical to its power stroke, even if the recovery stroke occurred at a lower angular velocity, the organism would exactly undo the work done by the power stroke. Something specifically tailored to the task of cellular locomotion is required. Thrust is generated by the continuous motion of waves through cellular projections attached to the main structure, flagella (on organisms called *flagellates*) and cilia (on *ciliates*).



"Various forms of *Dinoflagellata*. 2. shows the shell only; 4a is an undischarged, and b a discharged stinging-capsule; chr, chromatophores; fl. 1, longitudinal flagellum; fl. 2, transverse flagellum; l. gr, longitudinal groove; ntc, nematocyst; nu, meganucleus; nu, micronucleus; pg, pigment spot; t. gr, transverse groove." From Parker, TJ., *A Manual of Zoology*, New York, NY, The MacMillan Company, 1900.

Flagella are long tendrils that usually operate by actively propagating a helical or planar wave along their length from base to tip. Some forms of flagella operate purely by rotation at the base. In this case the tendril is left to drag freely through the fluid. The thrust generated by a flagellum is entirely due to viscous shearing and is generated parallel to the direction of wave propagation.

Ciliates are typically larger than flagellates although cilia are the shorter tendril type. Rather than propagating regular waves along their length, cilia act like miniature oars. Several thousand cilia may exist on the one organism, often arranged in closely packed rows across complete surfaces. Since at low Reynolds numbers a conventional rowing motion is unsuitable for propulsion, the cilia vary their rigidity during beating. The power stroke is made by pivoting from the base of the cilium. The structure itself remains relatively straight and rigid during this phase. On the recovery stroke, the cilium is allowed to relax along its length. Cilia are drawn tangentially through the fluid by a bend which rises from their base and passes along until it reaches the tendril tip. During this phase the drag on the tendril is caused almost entirely by the tangential flow of fluid across its surface. Consequently the recovery phase induces less resistance than the power stroke and the organism is able to use the cilia for locomotion by dragging fluid over its surface.

A frequently encountered natural solution to the problem of coordinating multiple limbs is the *metachronal wave*. This ensures adjacent propulsive structures operate slightly out of phase with one another to minimize interference and allow the continuous generation of thrust. Cilia are ideal candidates for control by metachronism. The direction and rate of beating may be fixed or under organism control.

Flagella and cilia are both amenable to modelling as rigid cylindrical elements connected to one another by springs in a mass-spring system, the subject of the next section.

Further reading

Dorin, A., Martin, J (1994), "A Model of Protozoan Movement for Artificial Life", in *Proceedings Computer Graphics International '94*, Melbourne, World Scientific, Kunii & Gigante (eds), pp. 28-38.

SECTION 2

Spring-loaded locomotion

TOPICS

1. Spring forces
2. Mass-spring systems
3. The spring-loaded worm
4. Sticky Feet

This *Bonga* automaton has a spring-powered gait of a more conventional kind. It is one of many similar machines from Kikkerland that bucks the trend of making battery-powered toys.

An articulated toy snake could conceivable have its rigid body-segments connected by springs along its sides, each segment being chained to its neighbours. As the rest-lengths of the springs are artificially reduced or lengthened, the springs would provide a force to flex the snake from side to side. If coordinated appropriately, they may cause the snake to slither. Cilia and flagella can be simulated similarly.

Spring forces

Robert Hooke was an English polymath who made the first observations of biological cells in plant matter (cork actually), which he illustrated in a significant work in the history of Science, *Micrographia* (1655). Hooke also invented the *balance spring*, a breakthrough in the regulation of watches and clocks, and he is known for his law relating the force on a spring to its displacement from rest length:

$$F = k\,(L - l)$$

The value k is known as the *spring constant*, a measure of the stiffness of the spring. L is the spring's natural length at rest, and l is its current length, for instance when compressed or extended. Hooke's Law is in fact only a first-order approximation of the force required to deform a spring by a certain length. But as long as the spring is not deformed very much and L-l is small, the law holds well, even for many deformable objects other than springs.

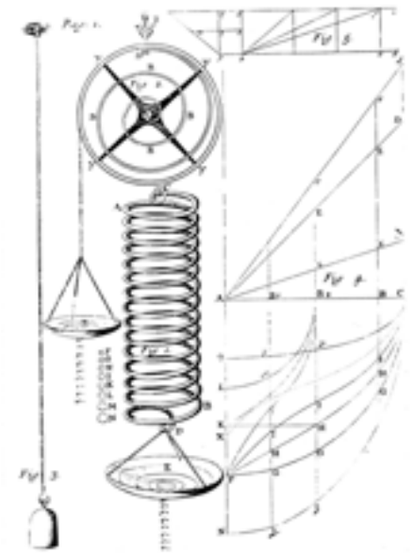
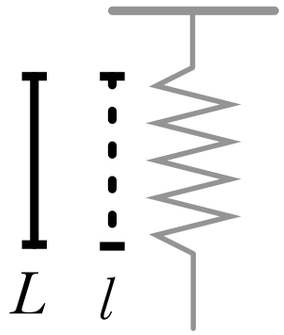


Image from Robert Hooke's *Lectures De Potentia* Restitutiva or of Spring Explaining the Power of Springing Bodies (1678).

Since the advent of spring-driven clocks, coiled springs of different sorts have continued to provide the motive force for automata. For software-based modelling, springs make suitable substitutes

for muscles. We can even build complete virtual organisms by modelling the rigid components of their bodies as point-masses or solid polyhedra, and connecting these to one another with springs that may introduce limb and torso motion of different kinds.

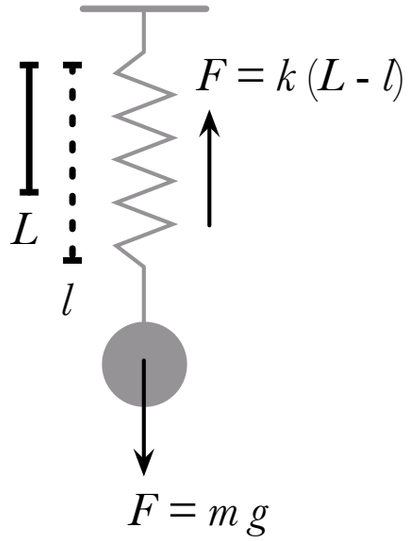


An idealised spring suspended from a fixed beam remains at its "rest length".

A simple mass-spring system

In the figure, an idealised massless spring is hanging from a fixed beam. The spring is at rest and its current length l is its rest-length, L.

When an idealised point-mass hangs at rest from the spring the spring stretches to a new l so that the force it applies is exactly counter to the force generated by the mass acting under gravity.



A spring supporting a hanging mass extends to a length beyond its rest length such that it counters the weight of the suspended mass.

If we now extend the spring even further by pulling the mass downwards, and we let go, the mass-spring system will enter simple harmonic oscillation. By extending the spring we force it to apply a counter force to that we generate. When we release the mass, the spring moves through its rest position into contraction. It then tends to expand again to the extended position. An ideal system that does not lose energy to friction will oscillate like this indefinitely but any real spring-mass system is *damped*. It loses energy in the form of heat generated by friction within the spring and by the movement of the mass through the air.

A damping force generated by drag of an object through a viscous fluid like air can be represented mathematically too. The damping force always acts in a direction opposite to the motion and is proportional to the speed of the movement and a constant accounting for the viscosity of the fluid and the size and shape of the object. This constant is the damping factor D in the following equation.

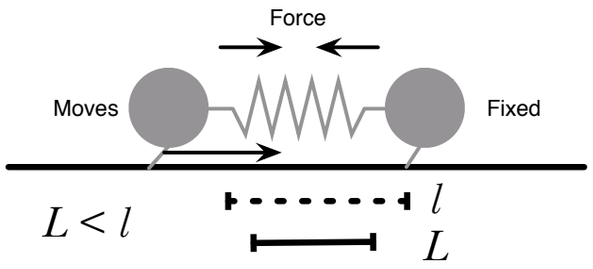
$$F = k\,(L - l) - D\,(dl / dt)$$

The equation states that the force acting on the mass is the sum of the spring force generated by its deformation from rest, and the damping factor multiplied by the rate of change of spring length (i.e. the mass' velocity). With a simple system like this it is possible to imagine a virtual creature.

The spring-loaded worm (William)

William is a simple, virtual, spring-mass construction involving two point masses, one spring and two claws. The claws constrain the motion of each mass to be in one direction only – they are infinitely grippy in one direction and slippery in the other.

The worm begins at rest. To make it move we artificially change the rest length of the spring. If we shorten it the spring



When the rest length is decreased so that it is less than the current length of the spring, the spring generates a force to return its ends to the new rest length, forcing the masses together. Only the left side is free to move in the required direction. The right side is fixed by its one-way claw.

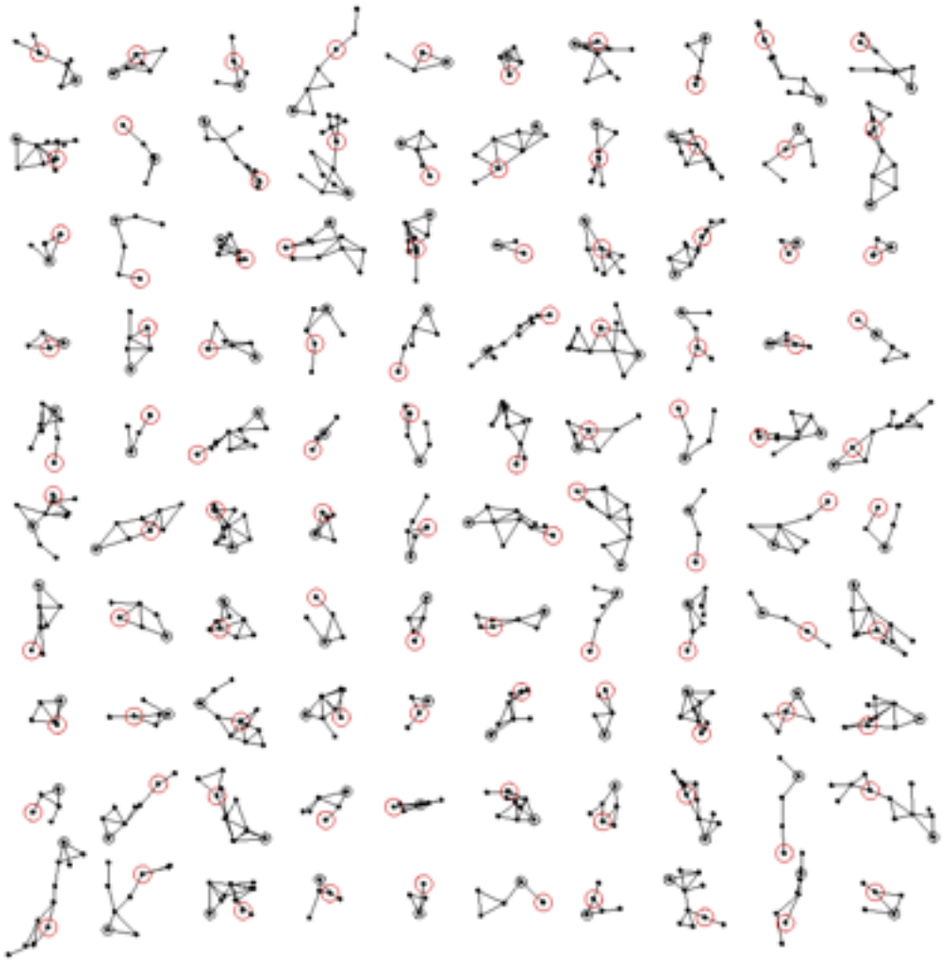
will apply a force squeezing the masses together. The right claw will hold one mass fast. But the left claw allows the other mass to slide to the right, past its resting position. Once it reaches the furthest right it can go, (the distance will depend on the initial value of *L-l*), it is prevented from returning under simple harmonic motion to the left by its claw. The spring is now compressed and applies a force pushing the masses apart. But only the mass on the right can slide in the direction required to allow the spring to return towards its natural length. It now slides right beyond the rest length of the spring, extending it. The spring now needs to contract but the mass on the right is held from moving in the required direction by its claw. This cycle repeats indefinitely as the two masses crawl, one following the other, towards the right.

We can simulate the movement of this creature using Hooke's law without worrying about the damping factor, but the resulting motion is uncontrollable. The addition of the drag force usefully causes the creature to come to a halt shortly after we stop animating the rest length of the spring.

To implement the physics required, we replicate the particle dynamics calculations we have already explored. I.e. from the sum of the forces acting on each mass we compute acceleration, velocity and positions in a series of incremental steps. The forces acting on the point masses are generated by the spring using Hooke's law. The surface friction acting on a mass *against* a claw is infinite and counters any spring force applied, but it is zero if the force acts in the opposite direction. The drag on each mass, just another force to take into account, is computed from the damping factor multiplied by the current velocity of each mass.

Sticky Feet

We needn't restrict ourselves to snakes and worms. Complex organism bodies can be animated using mass-spring systems as long as they can be decomposed into rigid structural elements and a set of stabilising and joint-activating springs. Everything from fish to humans has been attempted using this method. One novel application of mass-spring creatures involved the evolution of a whole virtual ecosystem by Greg Turk. Dubbed *Sticky-Feet* (for reasons apparent to anybody who has just considered the spring-loaded worm), hosts of creatures attempt to eat one another as they bumble around their 2D environment, feeling each other out with sensors. Successful predators reproduce offspring that inherit their traits, along with possible mutations that may (or may not) enhance their survival and replication. Evolving **virtual ecosystems** are considered in detail in a later chapter.

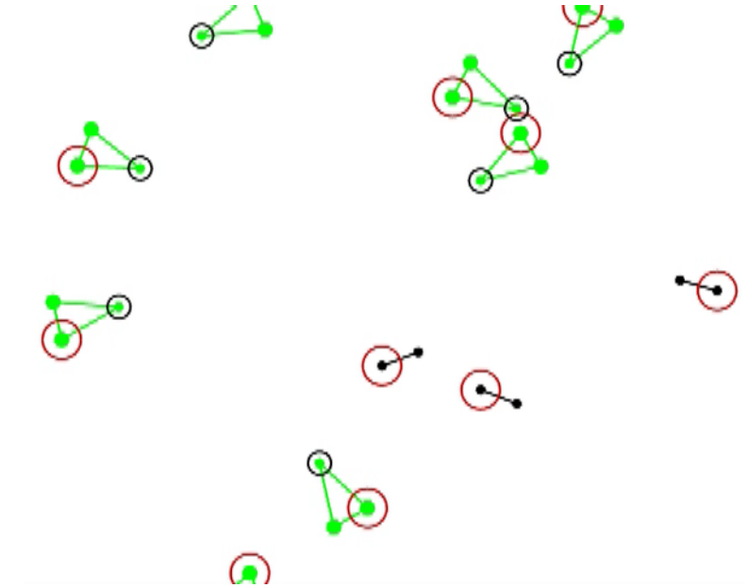


A zoo of evolved creatures from Greg Turk's *Sticky Feet*. Each virtual organism is an animated mass-spring system with a mouth (circled in grey) and a heart (circled in red) that it must protect from being eaten. Creatures also have sensors (not shown) that enable them to detect the presence of others. The sensors may dampen specific springs in their owner's body. If arranged properly this can cause the creature to turn towards (or away from) another. © Greg Turk 2010. Used with permission.

Further reading

Miller, G. (1988). The motion dynamics of snakes and worms. In SIGGRAPH '88 , pp. 169-173.

Turk, G. (2010). "Sticky Feet: Evolution in a Multi-Creature Physical Simulation", Artificial Life XII, Odense, Denmark, MIT Press, pp. 496-503.

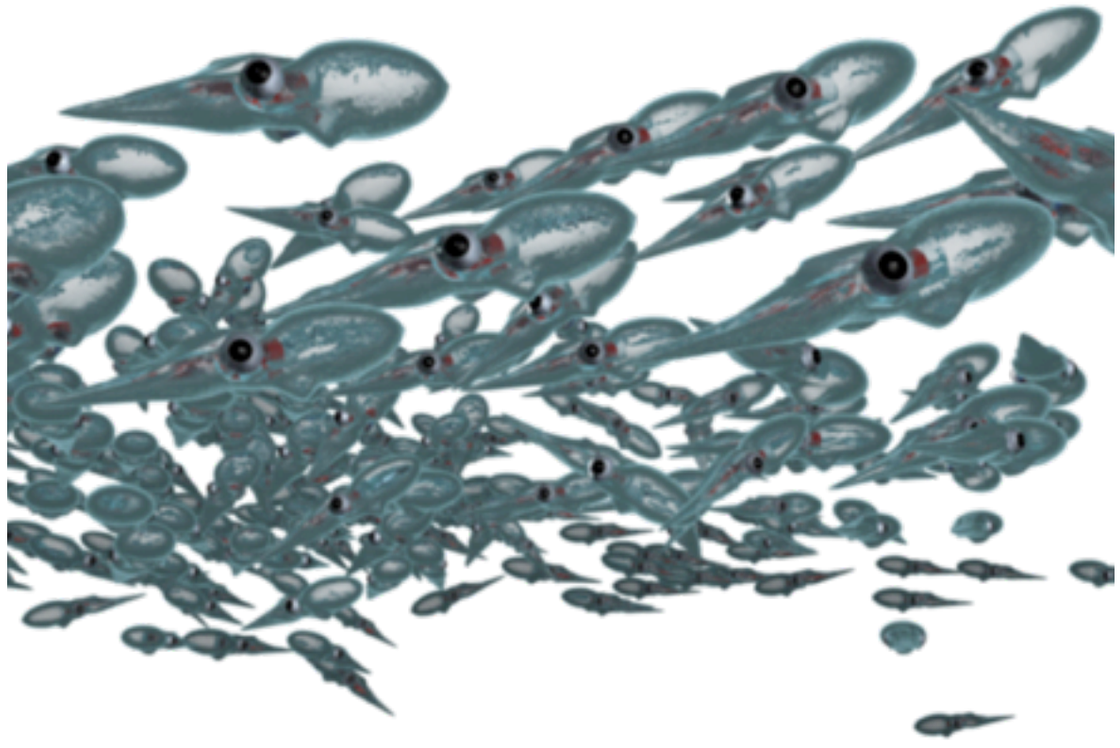


Sticky Feet interacting in an ecosystem.
© Greg Turk 2010. Used with permission.

Chapter 6

Group Behaviour

A daring and agile bird twisting and turning is exciting to behold. More breathtaking still is a flock in full flight as it wheels about the sky; a dark cloud, seething and writhing; a super-organism, one creature made of many. This chapter explores models of collective behaviour including flocks, herds, schools and swarms.



SECTION 1

Flocks, herds and schools

TOPICS

1. Aggregate movement in nature
2. Forcefield-based flocking simulation
3. Distributed flocking simulation (*boids*)
4. Additions to the distributed model

This group of squid is rapidly expanding in response to “fear” instilled by a massively repulsive vector.

Aggregate movement in nature

Birds and sheep flock, fish school and cows move in coordinated herds. One benefit of this is that it reduces the chance an individual group member may be singled out by a predator. These animals’ group behaviours share a number of characteristics. For simplicity we will refer to them all collectively as *flocking*. Flocking is a behaviour that:

- Is an aggregate, polarized, non-colliding motion of a group of animals.
- Propagates rapidly through a large group of animals.
- Can occur independently of the number of group members.
- Is complex and emergent from the interactions of the group’s individual members.

The emergence of a flock from the interactions of a group of animals has become one of the iconic phenomena of Artificial Life, along with the Cellular Automaton behaviour of The Game of Life discussed earlier. Prior to the current way of thinking about this behaviour which was pioneered by Craig Reynolds in 1987, a number of simulation models had been developed. These were primarily designed for computer graphics applications rather than for understanding biological behaviour. As soon as there are many agents in a complex, dynamic group such as a flock, manual animation becomes tedious and impractical. A procedural method for computing the behaviour of multiple creatures acting in concert is very valuable!

With the application for which it was developed in mind, it is interesting to explore an early model of flocking behaviour.

Forcefield-based flocking model

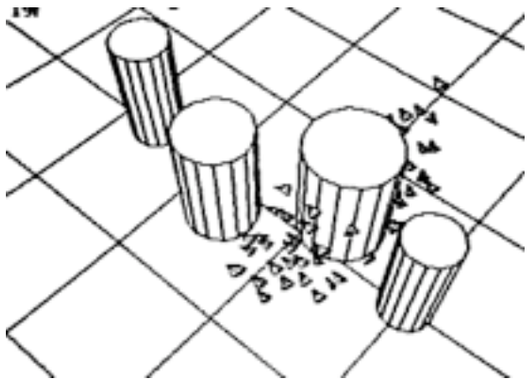
The *force-field model* of bird flocking behaviour was designed in 1985 by Susan Amkraut. She simulated a flock of birds in motion tests and a final animation entitled *Eurythmy* which was produced in collaboration with Michael Girard who focussed on legged animal motion (1985/89). Amkraut constructed repulsive forcefields around each static object and mobile bird in her animated scene. The forcefields assisted the birds to avoid colliding with one another and the architectural models. To choreograph the flocking behaviour, the birds were also subject to global forces that acted from sinks (attractive locations), sources (repulsive locations), spirals and orbits. These force-generating bird manipulators could be positioned and varied by the animator to create a dynamic forcefield. Paths for the birds were calculated automatically in discrete increments through the dynamic environment by taking into account the forces acting on each individual bird in its current location.

The force-field model is unconcerned with emergence. Although, with some caveats, the overall effect resembles real flocking behaviour, the model is implemented in a way that doesn't reflect the underlying capabilities or senses of individual birds. For our purposes, this must instead be based on each bird's local observations and internal state.

In some real flocks of birds an experienced leader guides the group on its migratory path. Models of this type have also been implemented, but if we are interested in the emergence of the flocking behaviour from the interaction of the flock members, we need to find an intrinsically distributed model.

Distributed flocking simulation

In 1987, Craig Reynolds published what has become the most well known flocking algorithm. Importantly from our perspective, he took the explicit step of treating each *boid* (bird-oid) as an individual agent among a population of others. In Reynolds' algorithm each individual reacts purely to its surroundings including static obstacles and other boids in close proximity. Each boid acts according to a set of simple behavioural rules shared by all flock members. Reynolds demonstrated his algorithm by animating a flock of birds and a school of fish in an animation titled *Stanley and Stella in Breaking the Ice* (1987). The principles his technique introduced have since become a staple of the animation industry being used not just for fish and birds, but even crowds of people and hoards of angry orcs, goblins and trolls.

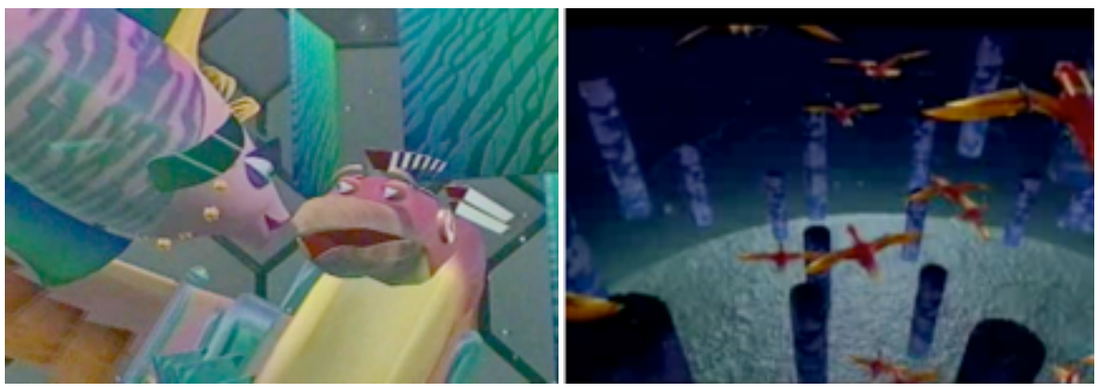


A screen-grab from Reynold's simulation showing a flock of *boids* (bird-oids) avoiding a collision with a series of columns by splitting into several groups while maintaining their general direction of travel and their proximity to other flock members. © Craig Reynolds, 1987. Used with permission.

The *boids* [Try the interactive widget online]

The boids of Reynolds' algorithm are best visualised as oriented particles – for effect, each should have a clear front so that the directional properties of the flock are clearly visualised. If the flocking model is to be constructed in 3D, clearly differentiated boid sides, tops and bottoms also add to the impact. Each boid in the flock is modelled as an autonomous agent that can perceive only its close neighbours and environment within a limited visual range. Within the limitations of its visibility, each boid repeatedly applies the same four simple rules every time it needs to fly a tiny distance forwards:

1. Avoid collisions with visible static objects
2. Avoid collisions with neighbouring boids
3. Match velocities with neighbouring boids
4. Move towards the centre of neighbouring boids

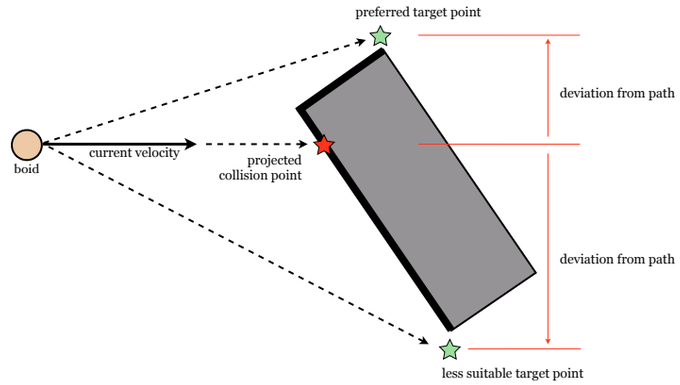


Still images from Craig Reynolds' *Stanley and Stella in Breaking the Ice* © Craig Reynolds, 1989. Used with permission.

By following these, each boid picks its way through space under its own guidance. Note that none of the rules explicitly specifies that a flock be formed. The flock emerges from the tendency to avoid collisions, match velocities and move towards a location surrounded by other boids. In addition, once it forms, there is no preferred direction of travel for the flock. Even the flock's path emerges from the individual behaviours of the boids. In a nutshell, here is how the individual rules work.

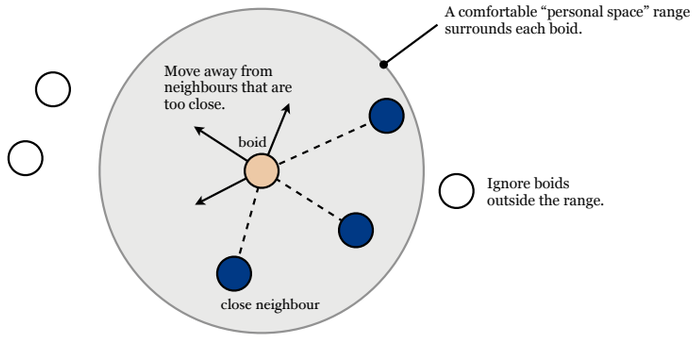
1. Avoid collisions with visible static objects.

Each boid is aware of its current position with respect to static obstacles around it and its own velocity. Every time it needs to make a decision about the next increment in its flight path, it computes any potential collisions it might have with static objects. If there is an imminent collision, the boid computes a location on the silhouette edge of the obstacle that doesn't cause it to deviate from its path too dramatically, and computes a force to direct it towards that location.



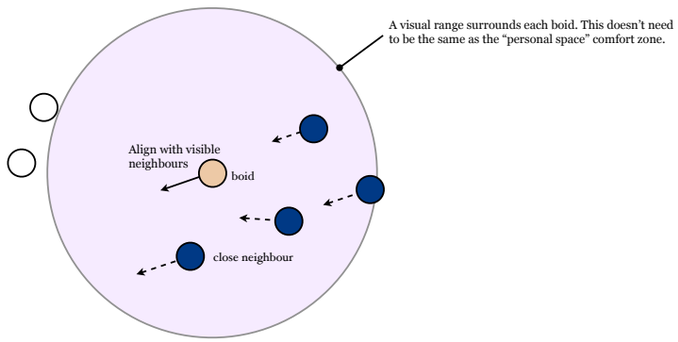
Static collision avoidance involves detecting imminent collisions and computing a location on the silhouette edge of the obstacle that will cause the least deviation from the desired goal or current path.

2. Avoid collisions with neighbouring boids.



To avoid colliding with neighbours, each boid computes vectors heading away from those that impinge on its personal space. The sum of these vectors will provide a direction that best moves the boid away from other flock members in its vicinity.

An easy way to avoid collisions with others is for a boid to keep its distance from them. This can be achieved by enforcing a region of protected space around each flock member. Within this space, forces (vectors specifying a direction and magnitude for the forces) can be computed to push a boid away from others much as was done in Amkraut's flocking model discussed previously. The explicit specification of a personal space range allows a boid to safely ignore flock members that are so far away as to be irrelevant or imperceptible.

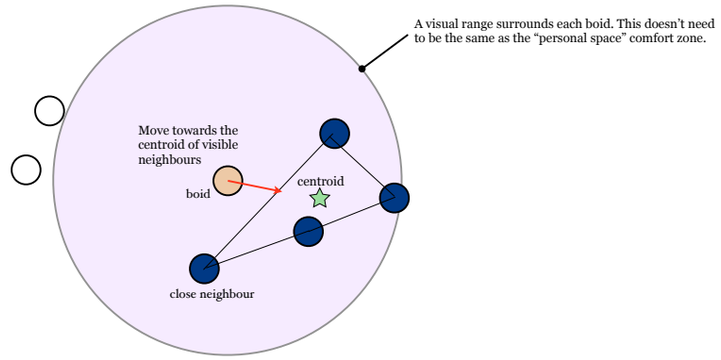


One way to avoid colliding with neighbours and to give the flock some direction is to have each boid steer in the same general direction as those around it.

3.Match velocities with neighbouring boids.

Another way to avoid collisions with neighbours is to align your velocity with theirs. This also contributes to the direction of travel for the flock. So, for each neighbour in the flock visible to a boid, it should try to match its heading and speed. This can be achieved by averaging the velocity vectors of the visible neighbours.

4. Move towards the centre of neighbouring boids.



To keep the flock together, the model provides a force directing each boid towards the centre of its local neighbourhood.

If the flock is to remain coherent, and not simply be spread apart by the repulsive forces specified at rule 2, then a tendency to move together can assist the velocity matching rule in maintaining boid proximity. This rule can be managed by identifying the local centre of each boid's neighbouring flock members (their centroid or average position), and computing a force to direct the boid whose movement is being computed towards it.

Once all of the forces have been computed for rules 1 to 4, these can be added to one another to compute a sum that accounts for the contribution of each rule. The components can be weighted as desired, for instance to make the collision avoidance more important than the velocity matching. Then the net force is applied to the boid, its acceleration is com-

puted from this, then its new velocity, and finally its new position. This series of computations matches that discussed for updating particle systems and in the section specifically on physical simulation. If all the boids in the model undergo this update process, over time a flock emerges!

Additions to the distributed flocking model

As noted earlier, a flock implemented according to a distributed model has a mind of its own. Suppose an animator wanted the flock to enter a room through a window, do one circuit, and then fly out through the door. How might this be controlled without destroying the emergent dynamics of the flock? One simple technique is to add a fifth rule to the boids' list that adds a small force causing each boid to head towards an animated location in space. This point *is* under the direct control of the animator. The resulting migratory urge can be used to pull the flock along while the flock members continue to observe rules 1 to 4 as well.

Suppose a shark was to appear in a tank of schooling fish. How might the school behave? By adding a highly weighted vector directed away from the predator to each boid's (*foid's*?) force, the desired rapid motion can be realised.

An additional consideration in modelling fish, birds, or even a swarm of flies, concerns enforcing limits to their motion. For instance few birds can hover. Perhaps non-humming-boids ought to have a minimum flight speed imposed. Likewise, maximum speeds, accelerations and rates of turn are worth implementing depending on the capabilities of the modelled organisms.

Further reading

Girard, M., Amkraut, S. (1990). "Eurythmy: Concept and process", The Journal of Visualization and Computer Animation, Vol. 1, No. 1, DOI: 10.1002/vis.4340010104, pp. 15–17.

Reynolds, C. (1987). "Flocks, Herds and Schools: A Distributed Behavioural Model", Comp. Graph. Vol 21, No. 4, (SIGGRAPH 87) pp. 25-34.

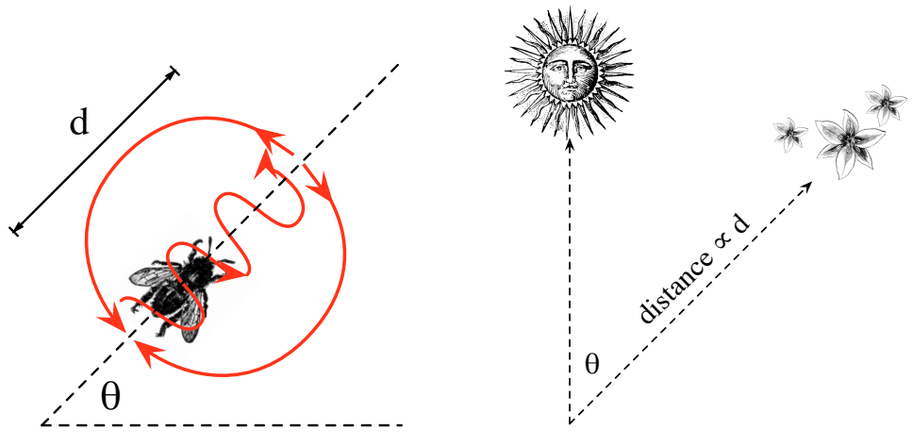
Tu, X., Terzopoulos, D. (1994). "Artificial Fishes: Physics, Locomotion, Perception, Behaviour", Comp. Graph. Proc. 1994, (SIGGRAPH 94) pp. 43-50.

SECTION 2

Swarm intelligence

TOPICS

1. Swarms in nature
2. Swarms and algorithms
3. How can a swarm build a heap?
4. Sequential and stigmergic behaviour
5. Wasp nest construction



A honeybee worker that has found a rewarding food source will conduct a *waggle dance* in a figure-eight (shown in red) when she returns to the hive to notify her fellow workers.

While traversing the central strip of the figure-eight she waggles herself. The length of this section of the dance is proportional to the distance to the food source. The angle with respect to the vertical of this phase of the dance indicates the angle of the food source with respect to the sun when leaving the beehive.

Swarms in nature

The idea of swarm intelligence arose largely from observations of the complex behaviour exhibited by the eusocial insects: bees, ants, wasps and termites. Colonies of these creatures exhibit more complex behaviours than "just" a swarm of flies or a school of fish. The special traits that warrant their description as highly social insects are: the reproductive division of labor by castes, the presence of at least two overlapping generations capable of contributing to colony labour, and the cooperative care of the young by the colony. The care of the young is usually carried out by a largely sterile worker caste that supports the reproductive individuals in the colony. In order to survive, these social insect colonies have evolved many unique behaviours that are emergent from the coordinated interactions of the individual colony members. This has suggested to many researchers that the colonies as a whole play the role of individual organisms made up of simpler organisms: they are *super-organisms*.

The bees' dance.

One behaviour that has fascinated natural scientists is the evolution of a complex dance language used by worker honeybees on returning from the hive to communicate the

Bees on a comb.

distance and direction of a food supply. Each dancer vies for the attention of her fellow workers, dancing more or less vigourously depending on the quality of the food source she has located. The orientation of her dance with respect to the vertical as she dances on a comb, relates the direction of the food source with respect to the current position of the sun. The number of "waggles" she executes in each repetition of the dance conveys the distance or amount of energy that must be consumed to reach the source. The net result is that the hive as a whole is a very efficient foraging entity, even when food is distributed widely.

The ants' trail. Ants are at an advantage over bees when indicating the path to food to their fellow workers: they are restricted to walk along solid surfaces. Rather than developing a complex dance language, they have therefore evolved a simpler system for designating a **pheromone** path between a food source and their nest. As each worker returns home bearing food, it deposits a scent trail along the soil, leaves, twigs or kitchen benches it crosses. The more workers that return along a path bearing food, the more strongly the path's scent will be reinforced, and the more attractive it becomes to unoccupied ants who join the busy foragers. As the food source becomes depleted, ants start to return to the nest empty-handed and therefore without reinforcing the pheromone trail. The scent naturally decays over time, dissipating into the atmosphere until it ceases to exist and ants are no longer recruited to visit the former food site. Some consequences of this method of path indication are that:

- Ants will choose the shortest of two paths of unequal length if the paths are presented simultaneously.
- After ants have chosen a path they are unable to switch to a new path, even if the new path is shorter.
- Ants will all choose one path in preference to half of them travelling a second path, even if the paths are of equal length.

Termite mounds dot the grasslands near Tennant Creek, Northern Territory, Australia. © Robert & Sara Dorin, 2009. Used with permission.

The termite's nest. The social insects are capable of collaboratively building astonishing architecture: complex nests with many chambers of different sizes and applications. Beehives, wasp and ant nests will be familiar to most people, but perhaps less familiar are the mounds built by

some species of termite, especially in northern Australia and on the African savannah. These can be several meters high and house literally millions of termites. In some species the mounds are oriented along a north-south axis to assist in thermoregulation. Vertical ventilation shafts that open out at the top and sides of the mounds are incorporated to assist in the movement of air through the structure. The nest, as well as having a substantial visible presence in the landscape, extends into the subterranean space. Passive climate control is vital for the individual termites working in areas where the daily temperatures may exceed 45°C (113°F) in the shade, it assists the colony to keep its brood alive, and in some species the climate regulation is also necessary to cultivate a garden of fungi.

Swarms and algorithms

The ability of swarms of social insects to "solve" complex problems collectively, especially when they forage or build their homes, has prompted researchers working with computation to scrutinise their methods. The hope is sometimes to learn about emergent phenomena of relevance to biology, but also to be able to apply nature's techniques to their fields, for instance in search, optimisation, town planning, art and architecture. This has lead to a computational science-styled approach to understanding biological swarms:

A swarm is a set of mobile agents which may: communicate with one another directly or indirectly; receive stimuli from the environment; alter the environment and perform collective or distributed problem solving.

In biological swarms the agents are usually heterogeneous as we have noted. They consist of different castes: queens, drones, workers; and possibly sub-castes such as foragers or soldiers. Each individual member is dependent on the colony as a whole for its survival. Many of the Artificial Life-based studies of these swarms focus on the behaviour of the workers in the biological colonies for inspiration. Hence, this will be the subject of the following sections also.

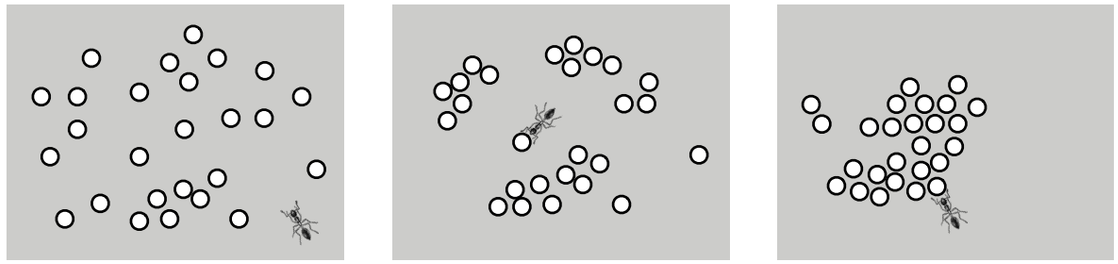
A basic requirement for interpreting swarm behaviour algorithmically is to decompose a problem, or structure to be constructed, into simple components and their local stopping configurations. One type of agent behaviour proceeds until a local stopping configuration is reached, triggering a new kind of behaviour in the worker. An action might be performed by an agent upon encountering a given environmental configuration or upon entering a specific internal state. The choice of action might be made deterministically or stochastically depending on the model. Different sets of rules can also be applied according to an external factor such as time or based on the state of an independent model such as one controlling climate. Entire sets of rules may be applied hierarchically depending on the success or failure of an individual agent's actions or on the conditions specified by the complex model external to the agent.

How can a swarm build a heap?

It is useful to think about deriving algorithms from social insect behaviour by considering a simple problem. Let's start this puzzle with a lone ant, capable of lifting one grain at a time. What intelligence is required for the ant to create a single heap of grains from a collection scattered across a plane? A seemingly sensible approach is for her to pick a location for the heap and move all of the grains to it. That isn't too difficult to imagine. But what if there are many thousands of ants and they need to collaborate? The added complication arises to communicate the location where each ant wishes to build, and the ants also need to agree on which of those is most suitable. Without coordination, some ants will be trying to build in one location, while others will be carting the grains away again for their own heap. An alternative strategy is to appoint a leader ant. Perhaps an ant of one caste might manage the heap-building behaviour of the others. That is certainly possible, but keeping in mind the properties of swarms that relate to emergent coordination, there is another possibility.

As we saw with the distributed model of flocking behaviour coordination can arise from the interactions of agents that independently follow the same set of basic rules in response to their local environmental conditions and state. This neatly matches the definition of a swarm just given above. Here are three heap-building rules for the ants:

- If you find a grain and your jaws are empty, grasp the grain and move or turn randomly a little;
- If you find a grain and your jaws are full, drop your grain and move or turn randomly a little;
- Otherwise, move or turn randomly a little.



Representations of three stages in the growth of a heap using the ant behavioural rules specified in the text.

These rules present a simplified model of nest wall construction in some species of ant. If they are carried out, even by a single ant, heaps gradually emerge since grains are being collected at random, but deposited only next to other grains. To begin with, small clumps of grain appear in the scattered collection. The larger these become, the greater the chance an ant carrying a grain will encounter them. Thus, the greater the clump size, the more rapidly it grows with respect to smaller clumps. This positive-feedback system eventually "runs away", resulting in a single large heap's dominance. Which fledgling heap will become dominant? The winner emerges randomly from the interactions of the grains and ants. The system has an *attractor* – a state to which it reliably converges – that consists of a single heap, but this is never really static. Ants keep bumping into the heap, picking up grains, walking around randomly and dropping them again. Nevertheless, these rules allow a collection of ants to create a heap, without

a leader performing centralised coordination and without any explicitly encoded form of agreement.

Sequential and stigmergic behaviour

In practice, the behaviours we find in nature are more stochastic than the rules just explored - not even an ant is a completely deterministic automaton. But in some cases insects do approach their day-to-day tasks very mechanically. Solitary female wasps of some species (e.g. the Australian *spider-wasp*) will, in the following reliable order: mate, locate a suitable site for a nest, build on (or clear out) that site, hunt for prey, sting and paralyse it, drag it back to the door of the newly fashioned nest, enter the nest to check it is clear, drag the prey into the nest, lay an egg on the body and seal up the nest from outside. Sequential behaviours like these are hard-coded into the repertoire of the insect. The local environment acts as a trigger for the insect to inform it that it is time to change its internal state from (trying to) execute one action to the next.

This solitary wasp has paralysed its prey and placed it beside the burrow. After it drags the grasshopper into the burrow and lays an egg in its twitching body, the wasp will block up the burrow entrance from outside with the gumnut (a seedpod from a eucalyptus tree). Photograph © Mani Shrestha 2014. Used with permission.

The hypothetical heap-building algorithm we explored in the previous sub-section is so restrictive as to disguise whether or not it is sequential, since once an ant had collected a grain of sand it was prohibited by its capabilities from collecting another. Apart from sequential algorithms, what else *could* an ant be following? An alternative is a *stigmergic* process, a term introduced in the 1950s by French biologist Pierre-Paul Grassé in reference to termite behaviour.

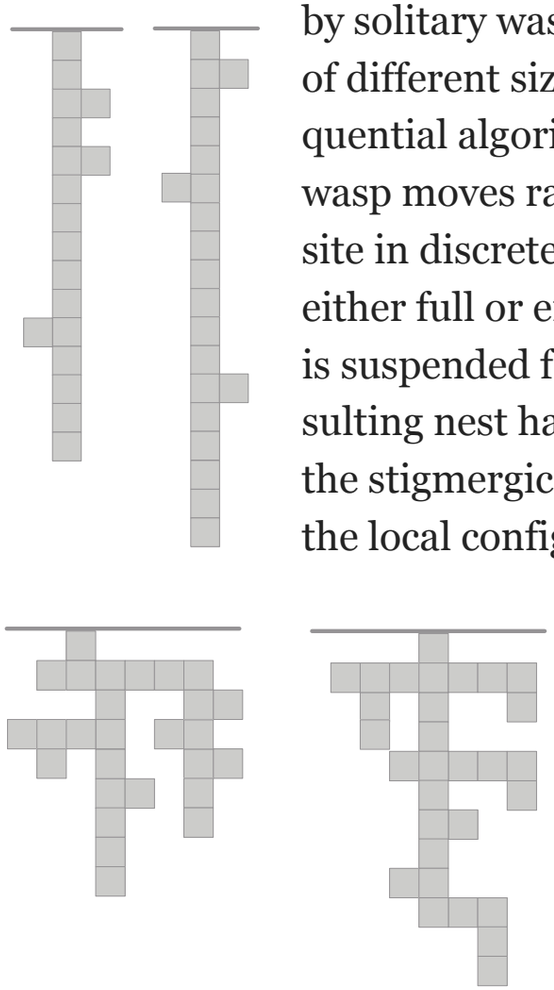
Stigmergy refers to behaviours, especially in insects, where the work performed on the environment up to a decision point informs the next action to be performed. Rather than operating from a fixed internal sequence, the state of the environment at any time as generated by previous building behaviour, will trigger insects to perform their next action. In this way it becomes possible for multiple workers to coordinate their activities since one worker may set up the environmental conditions from which any other can proceed. If instead each worker was following its own fixed internal sequence, such coordination becomes impossible. Under individual sequential control, each worker would have to step through its list, ignoring or destroying the work of its fellow colony members, in

Paper wasps' nest New South Wales, Australia. Paper wasps are social and live in small colonies of between 12 and 20 individuals. The adults feed on nectar and make their nests by mixing saliva and wood fibre to produce a paper-like building material.

order to bring the environment into the state its individual "current step" demanded.

Simulating wasp nest construction

Deneubourg, Theraulaz and Beckers implemented a simple simulation to test the kinds of nests that would be constructed by solitary wasps, or social wasps in groups of different sizes, following stigmeric and sequential algorithms. In their model, each wasp moves randomly across a lattice nest site in discrete time-steps. Each lattice cell is either full or empty of building material that is suspended from overhead so that the resulting nest hangs as if from a tree branch. In the stigmergic algorithm implemented, only the local configuration of a wasp is used to decide whether or not it will fill a lattice cell with building material. The action it chooses is based upon whether or not its current grid location is neighboured by one or two filled cells to its side, or a single filled cell above. In each of these cases a specified probability determines whether or not a new cell will be filled by the wasp.



Two sample nest structures generated by simulated wasps employing a *sequential* algorithm and two samples generated using a *stigmeric* algorithm with wasp groups of size 10. (Figures after Deneubourg *et al.* 1992)

The sequential algorithm the simulation implements has a wasp's current action influenced only by its past behaviour. The environmental configuration does not dictate current behaviour but it does act to permit or prevent certain behaviours – otherwise the wasp might unrealistically place building material in midair. At any time a wasp is either in horizontal or vertical filling mode. A probability is specified for transition between these modes after a successful building action is executed.

With a solitary virtual wasp executing the stigmergic algorithm, the researchers found the nests built on the lattice were generally short and simple. With 10 wasps following the algorithm, the patterns produced were slightly deeper, but also had multiple sub-trees sprouting along their length (much like the Paper wasps' nest shown in the photograph above). The sequential algorithm generated short nests with a solitary wasp, and long narrow nests with very short branches when 10 wasps enacted it.

Although these are very simple experiments, they offer as yet unrealised potential for constructing architecture from the bottom up. This is how human cities might grow in the absence of an overarching plan governed by town planning and enforcement of high-level constraints. Cities that grow organically are found today in the slums of many countries and such free-form organisation is typical too of many medieval European towns. Perhaps it might also be the way of the future?

Further reading

Camazine, S., et al. (2001). Self-organization in biological systems. Princeton, New Jersey, Princeton University Press, pp. 341-375.

Deneubourg, J-L, Theraulaz, G. and Beckers R., (1992). "Swarm-Made Architectures", Towards a Practice of Autonomous Systems, Proc. of the First European Conference on Artifical Life. (eds) Varela & Bourgine, MIT Press, pp. 123-131.

Frisch, K. von (1967). "The Dance Language and Orientation of Bees". Cambridge, USA, Harvard University Press.

Hofstadter, D.R. (1980). "Ant Fugue", in "Godel, Escher, Bach: An Eternal Golden Braid", Penguin Books, pp. 311-336.

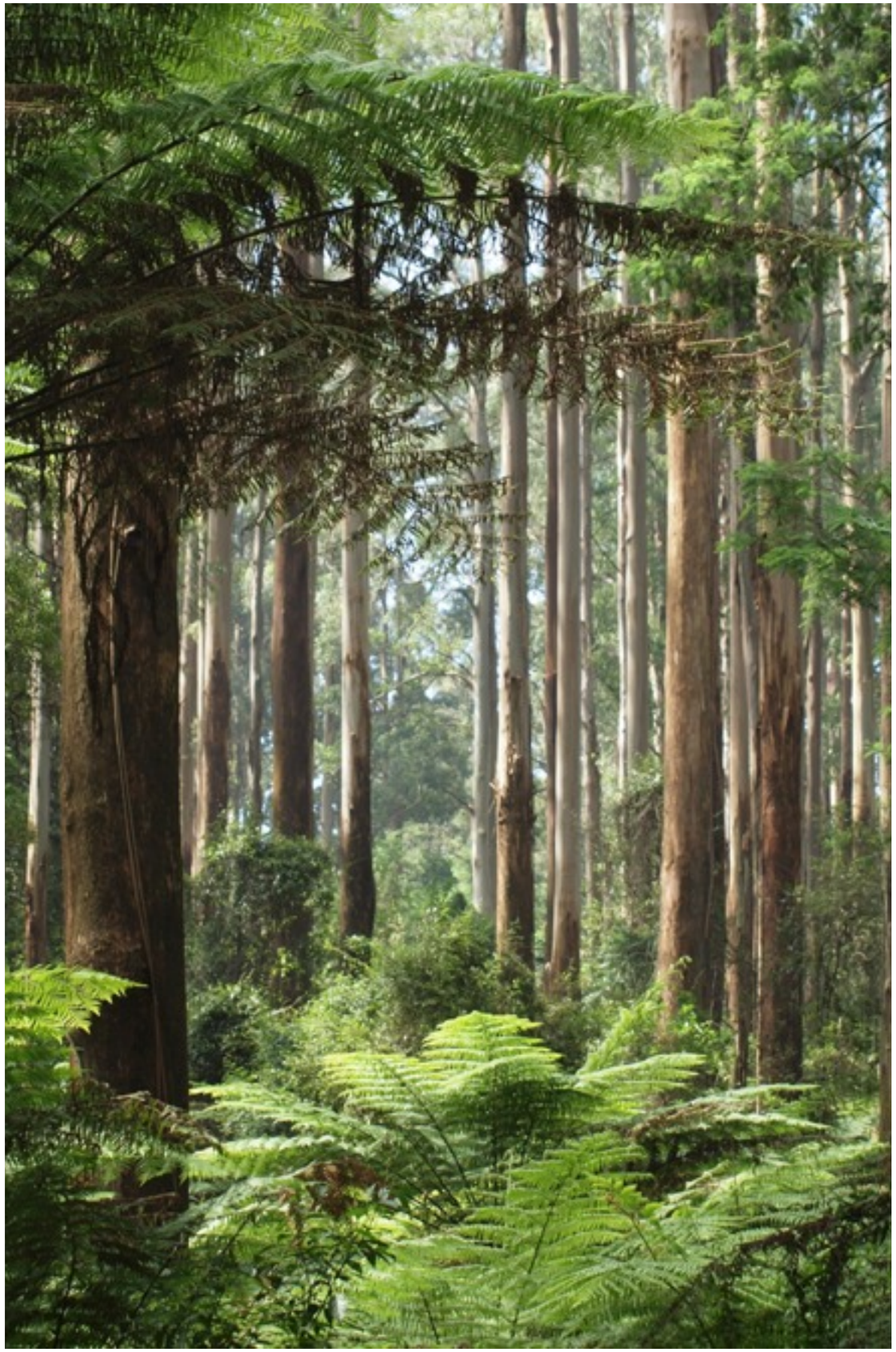
Millonas M. (1994). "Swarms, Phase Transitions and Collective Intelligence", Artificial Life III, (ed) Langton, Addison-Wesley, pp. 417-445.

Oster, G. F. and E. O. Wilson (1978). "Caste & Ecology in the Social Insects". New Jersey, Princeton University Press.

Chapter 7

Ecosystems

An ecosystem is a set of organisms, their abiotic habitat, and the interactions between all of these elements including exchanges of information, energy and matter. Ecosystems are intrinsically dynamic. This makes them wonderful systems to explore computationally, the subject of this chapter.

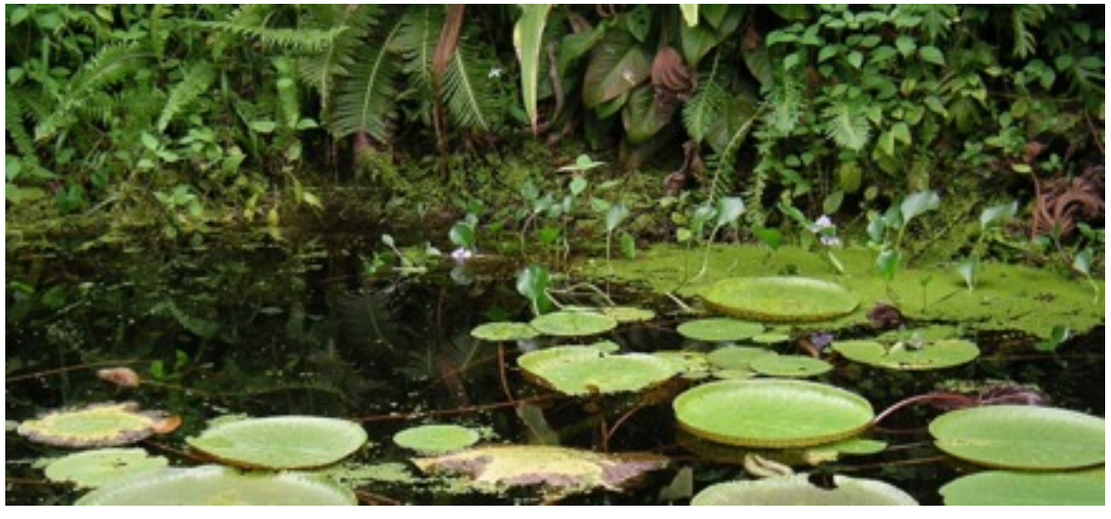


SECTION 1

Ecosystem models

TOPICS

1. The ecosystem concept
2. Daisyworld and homeostasis
3. The Lotka-Volterra equations
4. Individual-based models (IBMs)
5. Predator/prey individual-based model



Ecosystem is perhaps most clearly understood as an abstract concept. From this perspective, it is not something you can trip over as you wander around experiencing nature. Not only does it include plants, animals, earth, air and water, it includes the processes of material and energy exchange between all of the biological and non-biological entities in a region.

The ecosystem concept

From the perspective of an Artificial Life researcher, ecosystems are rich sources of emergent behaviours for investigation. The properties of individual organisms can only really be understood by reference to the dynamics under which their species has evolved. For instance predation strategies (pack hunting, chasing, trapping, ambushing) and predation avoidance approaches (camouflage, toxicity, running) are most often useful *between* species. Aggregate and social behaviours such as flocking, schooling, nest construction, communication and pack-hunting, develop within a species as coordination strategies for managing tasks such as predator evasion, foraging, or brood rearing.

Image from Ernst Haeckel's Kunstformen der Natur (1904), plate 26: Trachomedusae. Image in the public domain.

The need for a concept, the *ecosystem*, that unites invisible processes with tangible flora, fauna and abiotic entities is relatively recent. The development of the term *ecosystem* is therefore well documented and provides a useful reference to convey how the field of ecology has arisen. In addition, before discussing ecosystem techniques, it helps to know what is being modelled!

The German naturalist and artist Ernst Haeckel (1834 – 1919) coined the term "ecology" in 1866 to give form to the study of Natural History in the context

of Darwin's and Wallace's recently published ideas that organisms must struggle for survival. Ecology was to be the study of animals, their relationships amongst themselves, with plants and with the inorganic environment that affected their survival and reproduction. Sixty years later South African ecologist John Phillips, championing the view of another ecologist, Frederic Clements, insisted that a collection of plants and animals that had come into a harmonic relationship with one another and their habitat through succession to climax could be seen quite literally as a *Complex Organism*. Phillips viewed the process of succession as a kind of ontogeny, a process of development, for his own pet concept, *Biotic Communities*. His aim was, in part, to unify Botany and Zoology under a new banner.

> It is worth noting that Tansley was not concerned with *Systems Theory*, a field that came to the fore only after WWII. However his term's natural fit to this mould may be a part of the reason why the idea gathered popularity in the post-war years.

The English ecologist Sir Arthur George Tansley, unhappy with Phillips' argument, chimed into the debate and countered the use of *Complex Organism* by coining the word "ecosystem" in his retort, *The Use and Abuse of Vegetation Concepts and Terms* (1935). Several alternatives to ecosystem have been offered – biogeocenosis, microcosm, epimorph, elementary landscape, microlandscape, biosystem, holocoen, biochora, ecotope, geocenosis, facies, epifacies, diatope and bioecos – each with a slightly different slant. However in the UK, Europe, Australia, the USA and many other research communities, Tansley's term and its designated focus have stuck. This is true not only in science but also in politics, philosophy and even in marketing and popular culture.

Tansley's aim for the term was to give expression to a physical system that could legitimately take its place alongside those studied by Physics. Its components were animals, plants and abiotic material. He called attention to the significance of the exchange of materials and energy between organisms and the abiotic environment.

The preference for *ecosystem* by U.S. biologists Eugene and Howard Odum in the editions of their popular textbook *Fundamentals of Ecology* played a significant role in the term's post-war success. Eugene in particular refined his definition for the term across the three editions of his book. In the third (1971) he wrote,

Any unit that includes all of the organisms... in a given area interacting with the physical environment so that a flow of energy leads to clearly defined trophic structure, biotic diversity, and material cycles (i.e. exchange of materials between living and non-living parts) within the system is an ecological system or ecosystem.

With this text the Odum brothers arguably established the first generation of ecologists focussed on understanding the importance of the relationships between the Earth's biotic and abiotic components and the processes by which they exchange materials and energy. In particular, they drew attention to the emergence of:

Trophic structure – a hierarchy of interactions generated by organisms filling particular roles in food chains and webs.

Biotic diversity – a variety of different species, exhibiting a variety of behaviours and interactions with one another and their environment co-existing in a region.

Material cycles – the ongoing recycling of basic materials such as Nitrogen and Carbon through organisms and the abiotic environment.

Among the most interesting aspects about ecosystems from the perspective of Artificial Life studies, are their ability to self-organise in space and time and their ability to give rise to complex emergent behaviours at the level of individual organisms and groups of organisms.

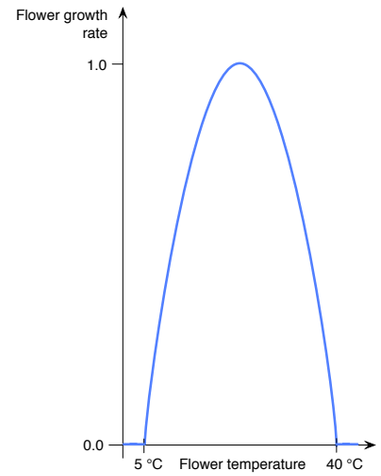
To an extent, these properties are due to the behavioural flexibility or adaptiveness of individuals. But it is evolution that has created this adaptability, and evolution that has been the primary source of the emergent properties of ecosystems through its action on species. Hence, we are left with a trio of drivers for the dynamics of ecosystems: energy exchange, matter cycles and evolution.

We will explore evolution later. In the remainder of this section we will explore ecosystems from the perspective of the interactions that give rise to trophic levels. These encompass energy exchange and material cycles. Models of the emergence of trophic levels, species interactions and diversity are valuable for making predictions about how our world will change in response to human activities. This allows us to plan our agriculture and manage natural resources. But ecosystem models are also valuable for the insight they provide generally in explaining why the natural world is the way it is.

Daisyworld and homeostasis

The lighter the colour of a daisy, the higher its albedo, the more incident energy it reflects.

The interaction between biota and the abiotic environment is one of the key factors moulding ecosystems. To take a simple example, the absence of water on some regions of the Earth's surface limits the kinds of organism that can survive there. These conditions give rise to different desert ecosystems. Organisms in the intertidal zones of the tropics however, have to contend with high salinity and regular inundation by sea water. The result has been the emergence of mangrove forests. The animals of the desert must all but eliminate the loss of water and salt from their tissues, while the animals of the mangroves must exclude excess moisture and salt. Both types of animal are *homeostatic* – to survive they must preserve the state of their



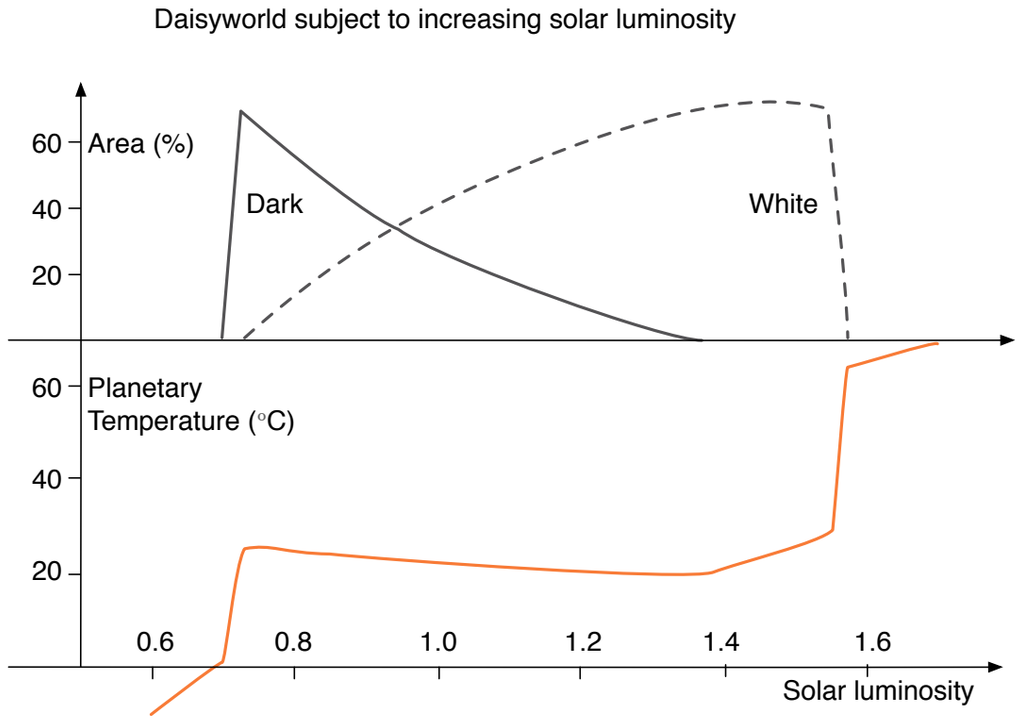
This curve represents the growth rate of daisies under different local temperatures. No growth occurs outside the range of 5-40°C. Growth is parabolic between the limits.

bodies within carefully controlled ranges – but they must do so under different adverse conditions.

In 1974, English environmentalist and scientist James Lovelock, and American biologist Lynn Margulis, published a paper suggesting that the entire biosphere was homeostatic; that the organisms on Earth maintained this homeostasis, ensuring conditions here remained suitable for life. This proposal, dubbed the *Gaia hypothesis*, later gave rise to a simulation model of how such a thing was possible.

Lovelock and Watson's model, *Daisyworld*, proposes an imaginary planet vastly simplified from the complexity of the Earth. On the planet grow two species of daisy of different colour. One daisy species is dark and absorbs more light than bare earth (it has a low solar reflectance; low *albedo*). The other species is lightly coloured and reflects more light than bare ground (it has a high albedo). The rate of growth of the daisies depends on their *local* temperature. This, in turn, is modified by the daisies via their albedo. The presence of dark daisies tends to increase the temperature of the entire planet over what it would be if its surface was bare. The reverse is true for the light flowers which reflect a lot of incident energy from the planet back out into space. Thus, the local temperature of a particular daisy is a function of the local albedo in its immediate vicinity, and the overall temperature of the planet.

The coupling between the rates of growth and death of daisies, the degree to which the populations of the two colours reflect energy or absorb it from the neighbouring star, the planet's surface temperature, and local temperature conditions for each daisy, form multiple interconnected feedback loops with interesting properties.



The lower graph shows the effect on the planetary temperature of increasing solar luminosity. Note that from a value of 0.7 to just below 1.6 units, the temperature of the planet is remarkably stable, even dipping slightly.

The upper graphs reveal the mechanism driving this homeostasis. Initially the population of dark daisies shoots up to warm the planet. As solar luminosity increases, the population of black daisies is no longer viable. They make their own environment too hot and are replaced in increasing numbers by white daisies. Eventually, even the white daisies find the planet too hot to maintain within a livable range and the surface temperature rises as we would expect in the absence of any daisies.

Graph redrawn from Watson, A.J., Lovelock, J.E., Biological homeostasis of the global environment: the parable of Daisyworld, *Tellus*, Vol. 35B, No. 4, 1983, pp. 284-289

The changes in abundance of the light and dark daisies are driven at a local level by the flowers themselves. If a region becomes too warm for daisies, white flowers have the upper hand since they assist to reduce the local temperature, and the planetary temperature with it, to within the bounds where they can grow. If many white flowers appear and begin to cool the local area too drastically, black flowers start to get the upper hand as locally they will do better by absorbing more of the incident radiation and warming the vicinity. Eventually a steady-state is reached.

What is potentially even more interesting than these adjustments to steady-state, is the emergence of planetary stability, even as the solar radiation on the planet increases! Lovelock and Watson subjected Daisyworld to an increasing amount of solar radiation. They found that the plants were able to maintain the planetary conditions very well – the system was homeostatic with respect to temperature. Without any daisies the planet's temperature would (of course) be expected to rise proportionally to the rise in solar luminosity.

While Daisyworld is insufficiently detailed to draw too many conclusions about earth's biosphere, for our purposes it highlights the extent to which the biotic and abiotic elements of an ecosystem may interact. Organisms are more than capable of changing their local environments in many and subtle ways. Their interactions with physical processes can be exceedingly complex and have unexpected consequences. We should keep this in mind, even when we consider a system as simple as two interacting species.

The Lotka-Volterra equations

Dingo: a free-roaming Australian dog. Image © Jarrod Amoore 2009, Sydney, Australia. Licensed for use under Creative Commons Attribution 2.0 Generic.

The Lotka-Volterra equations model the population dynamics (the rate of change of population size) of two species. A predatory species (perhaps a dingo) that eats a prey species (perhaps kangaroos). The equations as discussed here were developed independently in the early 20th century by Alfred Lotka and Vito Volterra. This simple two-species food-chain is an early biological system for which mathematical models were attempted. The equations are quite simple and worth understanding in detail.

$$\frac{dx}{dt} = x(\alpha - \beta y)$$

$$\frac{dy}{dt} = -y(\gamma - \delta x)$$

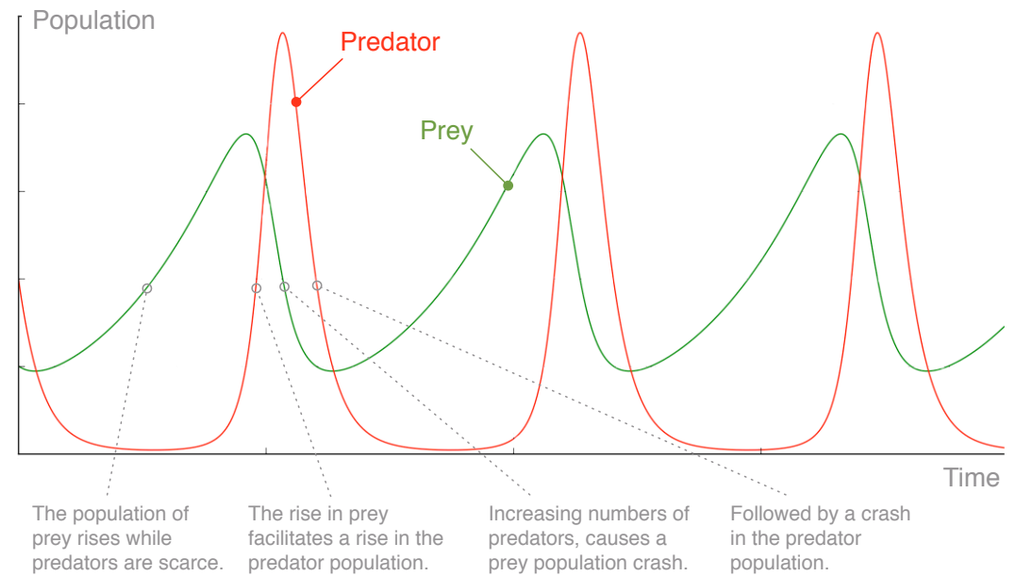
x - prey population size
y - predator population size
t - time
α - growth rate of the prey
β - rate at which predators consume prey
γ - death rate of predators
δ - rate at which predators increase by consuming prey

The rate of change of the prey population is represented by dx/dt. The rate of change of the predator population is given by dy/dt.

The expressions that represent the populations of the two species at any time are *coupled* differential equations. Each describes the rate of change of one population as a function of the current population of the other.

Hence the preys' population size x, increases proportionally to its current size times the growth rate α, but is reduced by an amount given by x times the number of predators y, multiplied by the rate at which predators consume prey β. Likewise, the expression for the predators' population size y, decreases due to its natural death rate γ, but increases by an amount computed as the rate at which predators multiply by consuming prey δ, and the number of available prey x. Overall, the coupling between the x's and y's in the equations reflects the mutual dependence of one species on the other.

For some values of the parameters, constant values of the populations may occur indefinitely. Perhaps more interestingly, as the graph provided indicates, for other values of the parameters, coupled cyclic oscillations occur. In a real biological scenario the patterns generated by the model could only appear if, no matter how large the prey population gets, there is always enough food to support them. Of course in practice the population size of prey is capped by the carrying capacity of their environment. In this simple model such things are not considered. Also absent from the model is any enforcement of



This graph shows the cyclic changes in population size of the predator and prey populations over time that result from α=0.04, β=0.04, γ=0.4, δ=0.2. The graph highlights the dependence of one population on the other.

minimum viable populations. The equations do not take account of this. Lastly, the model doesn't take into account any change over evolutionary time periods, for instance an arms race between a predator's ability to catch prey and the prey's ability to evade.

All the same, the simple model and Daisyworld are illustrative of the general principles of modelling ecosystems using mathematical and computational techniques. These are a very powerful paradigms.

Some limitations of equation-based models

The Lotka-Volterra equations give figures for the population size of the species of predator and prey over time. These are

numerical, aggregate properties of a species. To include grass in the model, a food source for the kangaroos, a new equation must be added to specify the relationship between the amount of grass at any time and the kangaroo population size it can support. The rate at which grass increases (its growth) and decreases (its consumption by kangaroos) must be represented. This adds to the complexity of the set of equations that need to be solved to understand the dynamic properties of the model and to use it for prediction. The more equations, the more difficult the system is to solve. In some cases there might not even be a feasible way to understand the system analytically.

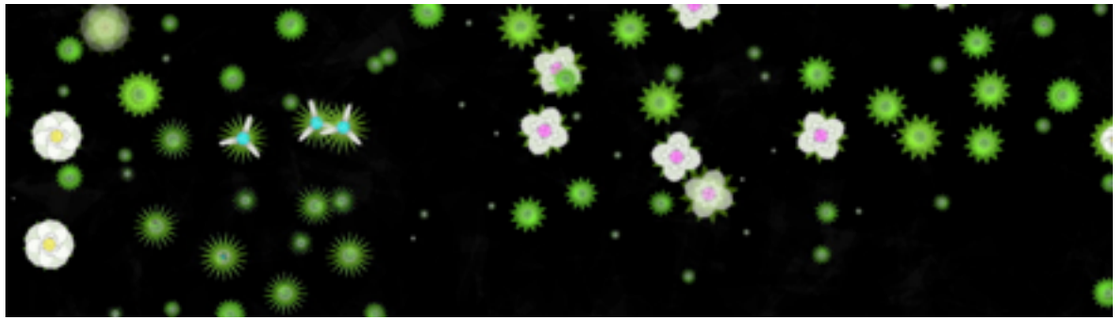
An additional aspect of working with aggregate properties in ecosystem models is that differences between individual predators and prey are not easily accounted for. For instance, all kangaroos are treated alike by the Lotka Volterra equations, as are all dingoes. Depending on the research questions of interest and the kinds of ecological phenomena under investigation, this may be inappropriate. For instance, many ecosystem phenomena are influenced by the spatial organisation of populations. Simple aggregate models treating all entities as spatially co-located will fail to represent this. Also, typically in nature there is a range of properties within a species. For instance some dingoes are better hunters than others; some kangaroos may evade predators successfully in some environments, and less well in other areas. Some kangaroos (the very young or very old) may just be slower runners than others. If these factors are significant for answering a particular research question, simple equation-based models may become unwieldy.

Individual-based models

Individual-based models (IBMs) are simulations capable of accounting for differences in behaviour, physiology and local conditions between organisms in an ecosystem by incorporating individual-level detail. These models are also known as *agent-based models* (ABMs) because each individual represented in the system is assumed to have the ability to act independently, based on its own properties, life history and local environment – i.e. each individual is an *agent*. A distributed model of flocking is an example of an individual-based model of bird behaviour. In this section we will explore a spatially explicit individual-based model derived from the predator/prey relationship explored through the Lotka-Volterra equations.

How do individual-based models surmount difficulties of equation-based models?

Individual-based models include data-structures that represent relevant organisms individually. For example, a simulation of bee/flower interactions might include representations of each and every bee in a hive and each and every flower in a foraging patch. Each flower would conceivably have properties that include its location in space (space is often modelled explicitly), colour, size and scent. Each bee might store a unique memory of which flowers it has visited in the past, which of those had contained nectar, their colours and locations. Agent-bees could have physiological traits modelled too



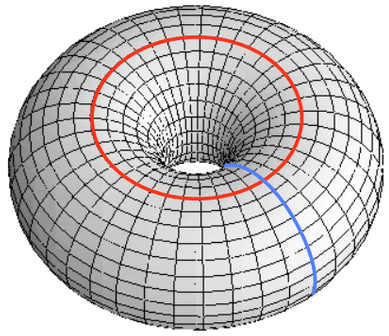
Here is a visual representation of a spatially explicit individual-based model of the interaction between flowering plant species. In this model, flowering plants consume resources from their environment and provide new ones that potentially allow new species to survive where they otherwise might not. Upon reaching maturity, each flower (asexually) reproduces by randomly scattering seeds in its vicinity. If they happen to land in a suitable location and all resources they need to survive are present, they in turn bloom. Otherwise they decay. (Constellation (2009), generative electronic-media art installation.)

so that individuals had a current position in the modelled space, and a velocity, current nectar load and maximum capacity.

The purpose of such models is to set up a virtual ecosystem with all the relevant organisms represented, including behavioural rules that will govern how they make decisions on a moment-by-moment basis. The rules are akin to those explored in the distributed flocking model.

Once the initial conditions and rules are set up, the system is left to run itself! The researcher then examines the system's behaviour to determine how closely it matches known data. The model might be revised multiple times before it correctly represents the empirical evidence in the desired way and to the desired level of accuracy.

After validation, experiments may be conducted using a simulation by varying parameters and procedures. The results can be analysed as they would be in any regular field experiment. The advantage simulations hold over field experiments is their flexibility. What if the flowers were bigger? What if the bees were faster? What if it started to rain? How would the system behave if the density of flowers was halved? These kinds of questions can be answered relatively easily in simulation, even if equivalent real-world experiments would be impractical or well-nigh impossible to conduct.



A torus world can be generated from a rectangular sheet by curling the left side of the sheet around to the right side to make a tube. (The red line in the figure shows the junction.) And then the top side is curled around to meet the bottom side so that the tube becomes a torus. (The blue line shows the junction of the tube ends.)

Even though this world is conceptually a torus, it is usually just visualised as a rectangle on which a creature moving off the top edge "miraculously" appears back in the world at the bottom, and *vice versa*. Likewise for the left and right edges.

A predator/prey individual-based model

In this section we will describe one way to build an individual-based model that extends the simulation of dingoes and kangaroos just described by the basic Lotka-Volterra system.

Space. A common way to model space in individual-based ecosystem models is as a rectangle. It is also possible to imagine a torus (doughnut) world. This may seem

strange but it is useful since a creature roaming the surface of a torus encounters no boundaries and therefore the world does not introduce edge effects into the simulation. For instance, on a torus a kangaroo cannot be trapped against an artificial world edge by a dingo. Unless we specifically wanted to model hard boundaries, a torus is probably best for our application.

The space may be continuous, allowing agents to occupy any floating point (x, y) coordinate. In our case we will use a grid-based space tiled with square cells. Each cell is given an integer Cartesian coordinate identifier. At any time during a simulation run, each cell may contain either dirt or grass, and possibly a kangaroo and a dingo too.

Next we design classes of object to represent entities in the simulation space. These will have a number of parameters to be updated in discrete time-steps as a simulation proceeds. Individual agents will undergo addition (+) to the simulation, transformation (~) within it, and removal (-) from the simulation as described for each class of entity below.

Grass. (+) In our simulation, (green) grass will grow in patches, one grass patch per grid cell. Hence each grass patch will require an (x,y) grid location. All new grass agents are created at the start of a simulation.

(~) A grass plant in a cell has a single state. It is either eaten or edible. An eaten grass plant will return to the edible state with a fixed probability in every time step.

(-) A grass agent is removed from the simulation when a kangaroo occupies its grid cell and eats it leaving only (brown) dirt.

Kangaroos. (+) New (grey) kangaroo agents appear in the simulation if an existing kangaroo agent gives birth to them. A kangaroo gives birth to a new one with a particular probability in every time step.

(~) Kangaroos are mobile agents. They require a grid-cell position that may be updated every time-step as they move, randomly in this simple model, from one grid cell to a neighbouring cell. Every time step they are alive, a kangaroo consumes a unit of its energy. A variable is needed to store each kangaroo's current energy store. A kangaroo in a grid cell occupied by grass will eat the grass and gain a unit of energy in its store.

(-) When a kangaroo runs out of energy, it is removed from the simulation due to starvation. When a kangaroo occupies the same grid cell as a dingo, the kangaroo is removed from the simulation due to predation.

Dingos. (+) New (red) dingos appear in the simulation when living dingoes give birth to them. A dingo gives birth to a new one with a particular probability in every time step.

(~) Dingos are mobile agents. They require a grid-cell position that may be updated every time-step as they move, randomly in this simple model, from one grid cell to a neighbouring cell. Every time step they are alive, a dingo consumes a unit of its energy. A variable is needed to store each dingo's current en-

ergy store. A dingo in a grid cell occupied by a kangaroo will eat it and gain a unit of energy in its store.

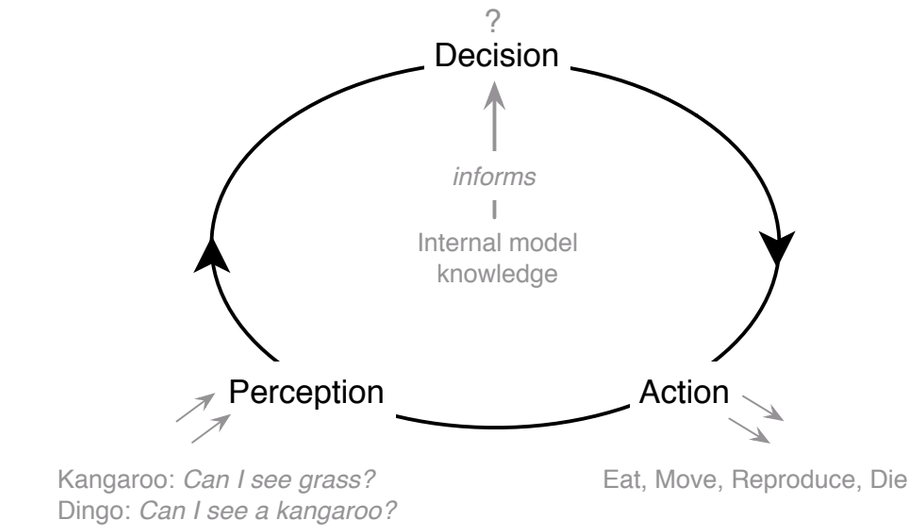
(-) When a dingo runs out of energy, it is removed from the simulation due to starvation.

Running the simulation
[Try the interactive widget online]

Now the simulation is run and the results are analysed, firstly to verify that the software is running as it should, then to validate the behaviour of the model against empirical data and expert opinion. And then, with luck, to explore the dynamics of the ecosystem under conditions where empirical data is unavailable.

During a run, the simulation state is updated incrementally. Every time-step, the agents are updated one at a time in a random order. Random ordering of the agent update is one way to avoid granting some agents a consistent head-start. Each active agent examines its local environment, for instance a kangaroo must check for grass in its cell. If grass is present it may decide to eat it. The eaten grass agent would then have its state set to *eaten*.

Every dingo checks for the presence of a kangaroo in its grid cell. If none are present, one dingo might decide to move. It selects a random cell in its neighbourhood and executes the action to move there. Another dingo might test its reproduction condition and generate a new dingo in the simulation. A



Each agent updates itself once every time-step. It usually begins by looking in its environment. Based on what it sees and its internal state, it then makes a decision about what to do. Lastly, it attempts to execute the action.

different dingo may have no energy reserves. It will die of starvation and be removed from the simulation.

As the simulation unfolds, data about its performance can be examined, or collected for later analysis. In this case, since we are interested in extending the system modelled by the Lotka-Volterra equations, it makes sense to plot the population size over time as we did before. It is often helpful to provide interactive control over the parameters of the simulation such as the probability of grass regrowing and the birth rates for the kangaroos and dingos. Experimentation with these values as the simulation runs can provide a good feel for the dynamics of a system: How sensitive is it to various parameters? What parameter values generate stable systems? Which parameter ranges cause oscillations in system dynamics? Which cause a crash?

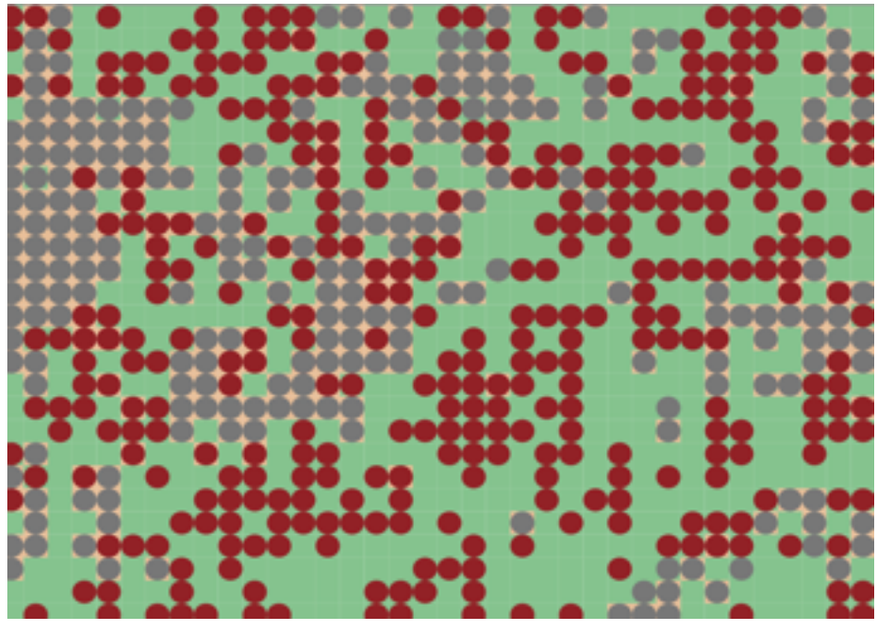
Further reading

Dorin, A., K. B. Korb and V. Grimm (2008). Artificial-Life Ecosystems: What are they and what could they become? Eleventh International Conference on Artificial Life. S. Bullock, J. Noble, R. A. Watson and M. A. Bedau, MIT Press: 173-180.

Grimm, V. and S. F. Railsback (2005). "Individual-based Modeling and Ecology", Princeton University Press.

Watson, A.J., Lovelock, J.E., (1983). Biological homeostasis of the global environment: the parable of Daisyworld, *Tellus*, Vol. 35B, No. 4, pp. 284-289.

Wilensky, U., Reisman, K. (1999). Connected Science: Learning Biology through Constructing and Testing Computational Theories – an Embodied Modeling Approach. *International Journal of Complex Systems*, M. 234, pp. 1 - 12.



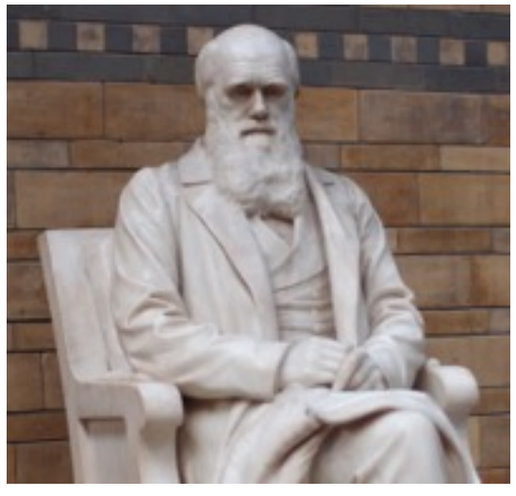
Screen grab of the described predator/prey simulation.

SECTION 2

Evolution

TOPICS

1. Evolution by natural selection
2. The fitness landscape
3. Aesthetic selection
4. Genotypes, phenotypes and fitness functions
5. Automated evolutionary computation



Charles Darwin looks thoughtfully out into space at the Natural History Museum, London.

Portrait (detail) of Alfred Russel Wallace by J.W. Beaufort, 1923. British Museum, London.

Evolution by natural selection

Throughout history, at least as far back as the ancient philosopher and poet Empedocles (c. 492–432 BCE), it has been repeatedly suggested that humans were derived from ancestors in an evolutionary chain. Many attempts have been made to explain how and why this might have happened, and to present evidence in support or counter to the claim. Until the 19th century no satisfactory mechanism to explain the evolutionary process had been documented.

In 1859, Charles Darwin and Alfred Russel Wallace published their theories of evolution by natural selection and the world has never been the same since. The importance of their work to science, and culture generally, can hardly be overestimated. Why? Because the theory of evolution by natural selection proposes a concrete mechanism for the production of the immense diversity found in nature. It explains the distributions of species we find now, and the lineages recorded in the fossil record. It explains many of the interactions between organisms within and between species. It explains their degree of specialisation in morphology, much about their behaviour, and their adaptedness to different environments. The theory does not invoke supernatural beings or events, it relies on basic physical principles. The theory can be used to make predictions, and its implications can be tested.

Evolution by natural selection has a few conditions required for it to become established. It requires that:

- There are differences between individuals of a species.
- Many of the differences between individuals in a species are inherited by offspring from their parents.
- Some of the differences enhance survival and increase the number of offspring an individual successfully produces.
- Differences in survival and numbers of offspring between individuals with different traits will result in variation in the numbers of descendants bearing their parents' traits.

...consequently, variations that allow an organism to be more successful at producing offspring will tend to accumulate in the population. Variations that lower reproductive success will tend to be eliminated from a population.

While the theory can be quite simply stated, its consequences are far-reaching and not at all easy to understand. After a hundred and fifty years the implications of evolution in different contexts are still being thrashed out by many thousands of scientists on a daily basis. Just to be clear: it is *not* disputed within science that evolution occurs and that it has shaped life. But the effects of the evolutionary process are not always easy to understand and the future implications of the process can be difficult to predict.

The following sections explore evolution with the aim of understanding its implementation using computers, an idea that appears to have begun in 1954 with the work of Nils Barricelli in the USA. In this text, our interests relate specifically to digital evolution and Artificial Life. The technique has been used more widely for engineering and optimisation problems also, but these areas won't be of direct concern to us here.

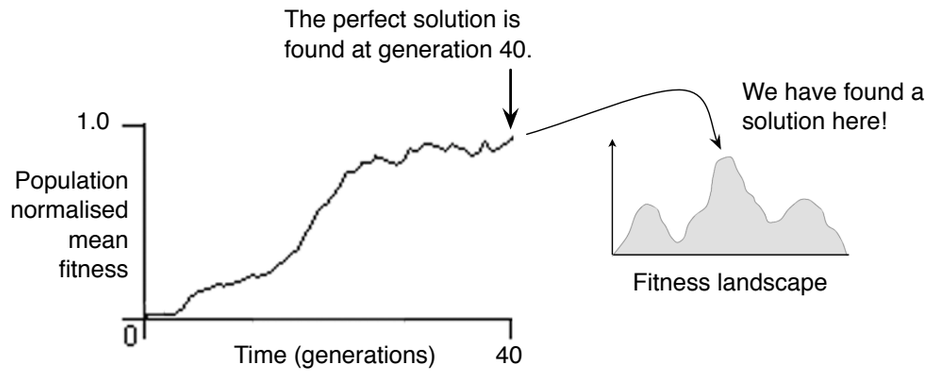
In order to reach the point where the principles of digital evolution will be easy to follow, it is first useful to explain the concept of the fitness landscape. After that a concrete example of digital evolution will be presented.

The fitness landscape

At one level, evolution can be understood as a search or optimisation algorithm. When applied to a "problem" (e.g. survival in a particular environment), this algorithm can make incremental improvements to existing poor "solutions" (e.g. organism morphology, physiology and behaviour) to seek out currently undiscovered superior solutions. The quality of a solution is referred to as its *fitness*. Higher fitness implies greater reproductive success.

Fitness can be thought of as a score that a solution achieves when tested in a particular environment. For example, a quick rabbit will evade more foxes than a slow one. The faster rabbit is therefore likely to foster more children that inherit its leg speed than the slow rabbit; provided both live in an environment where foxes pose a threat. A spider with a sticky web may catch more insects than its neighbour whose web is not sticky. The sticky-webbed spider is therefore more likely to be

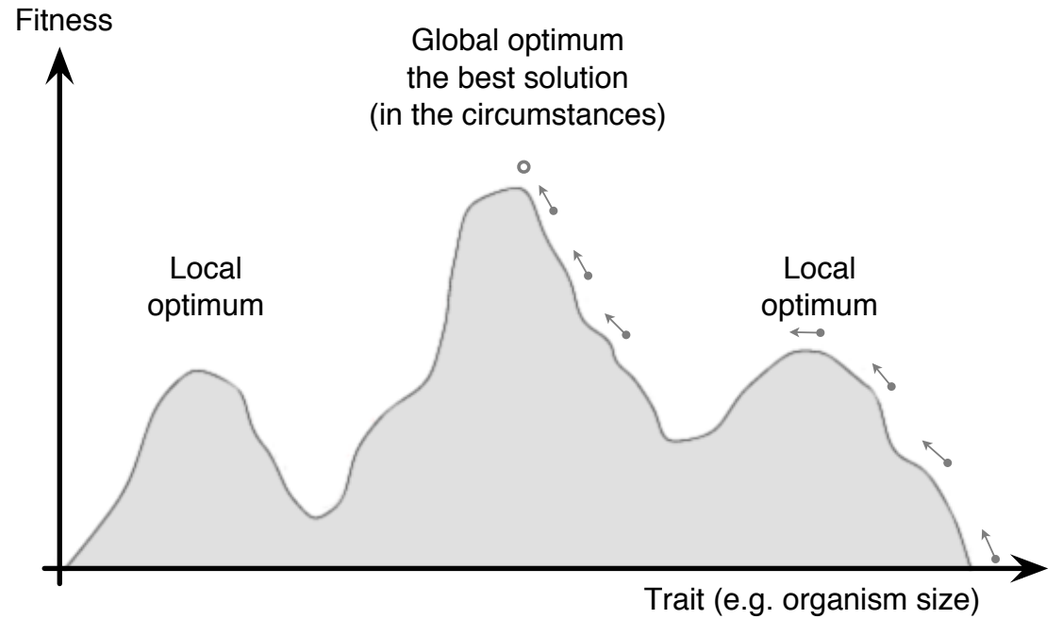
better nourished and able to produce more offspring that inherit its sticky-web trait than its neighbour.



A fitness vs. time plot for an imaginary evolutionary process within a stable environment. At generation 40, one or more members of the population are born that exhibit the optimal value of the trait(s) depicted in the fitness landscape.

Of course in nature nothing is so simple! A fast rabbit might also be bright pink and its slow friend may be well camouflaged. The sticky-webbed spider might be tasty to birds and its neighbour might be foul. Many traits of an individual decide its overall fitness in a particular environment. This, as well as the existence of countless different, dynamic environments, is part of the reason why there is such a diversity of life on earth.

Imagine a single trait that influences the fitness of a species in a particular environment, in a complex way. For instance the size of an organism will influence its maximum acceleration and deceleration in a chase, the amount of energy it can store over winter, the amount of nutrition it needs to survive, the rate at which it cools down and heats up, its success in a fight,



This is a sketch of an imaginary fitness landscape. Different values of the trait, as might occur in a population of organisms, generate troughs and peaks of fitness giving the appearance of a landscape across the space of possible organisms. The small arrows indicate the desired path of a population's maximum fitness: rising from an initial low value corresponding to a poor choice of trait, over a local optimum and trough, and onto the global optimum which represents individuals with the best choice of trait given the circumstances.

The idea of generating a fitness landscape showing the fitness resulting from all possible combinations of genes harks back to a 1932 paper by American geneticist Sewall Wright. His landscape was set out as a 2.5D topographical contour map in gene space.

its ability to fly and its attractiveness to mates. All of these factors have the potential to influence reproductive success. If we plot on a graph the fitness of (potential) members of the species exhibiting a range of values for the trait, we generate what is known as a *fitness landscape*.

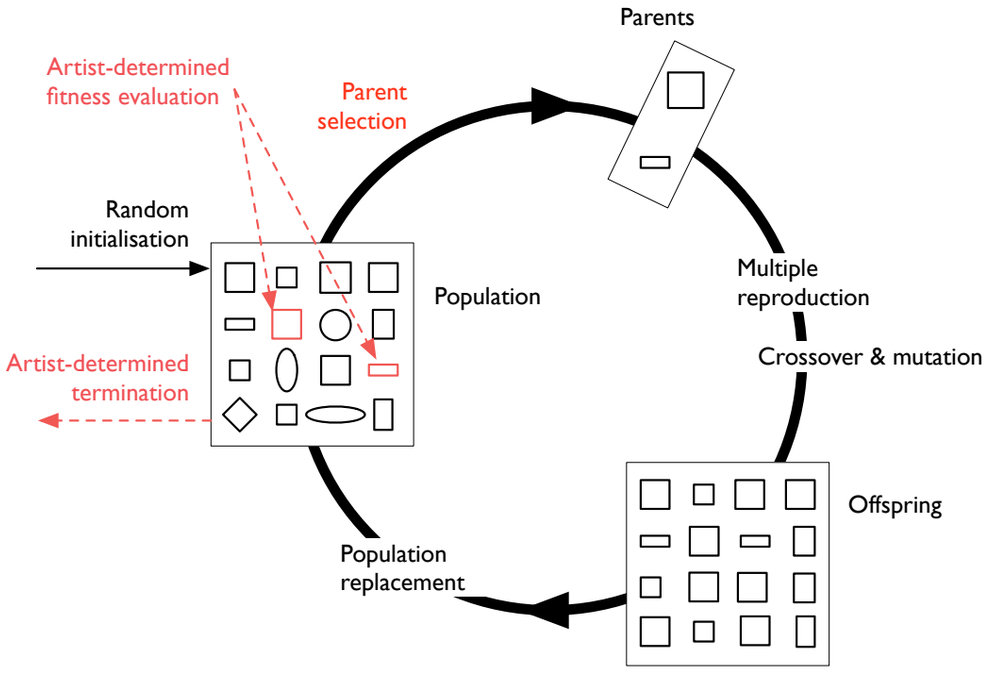
For evolution to work successfully, it must be possible to incrementally improve the fitness of the members of a population

by edging slightly up the mountains and skipping over the troughs of the fitness landscape. In a landscape that resembles a single spike in the middle of a vast plain, the required gradual climb to the peak is unavailable. There will also be no smoothly inclined path in a random jagged fitness landscape. The evolutionary process doesn't work properly under either of these conditions. It is important to keep this fact in mind, especially when evolutionary processes are being used in software.

Aesthetic selection

This section describes how the evolutionary process can work in software to generate visual forms that are manually selected by a user and automatically bred by the computer. One way to conceptualise the visual art-making process is to consider an abstract space containing all possible images – an infinite library of all potential works of visual art. From this unusual perspective, the artist's job is to search the infinite library for a desirable picture. Like any fitness landscape, the space of possible images contains areas of low and high fitness. The fuzzy boundaries of these regions will be dictated by personal taste. How might such a truly vast space be searched within the lifetime of an individual artist?

Evolutionary software can be used to assist humans to "find" visual art, architecture and industrial designs from a space that is so huge it is beyond the capability of humans to even imagine. The aim of the approach is to allow artists and designers to do what they are good at (assessing interesting, relevant or beautiful designs and features) and to use computers for what they are good at (crunching vast arrays of numbers, generating complexity and novelty).



An outline of the artificial evolution of shapes using purely aesthetic criteria for fitness. The process begins with the random initialisation of a population of forms from which the user manually selects those that are, for whatever reason, preferred. These are used as parents to breed a new population – just like breeding pigeons, apples or roses, only the computer does the form-mating, not biology.

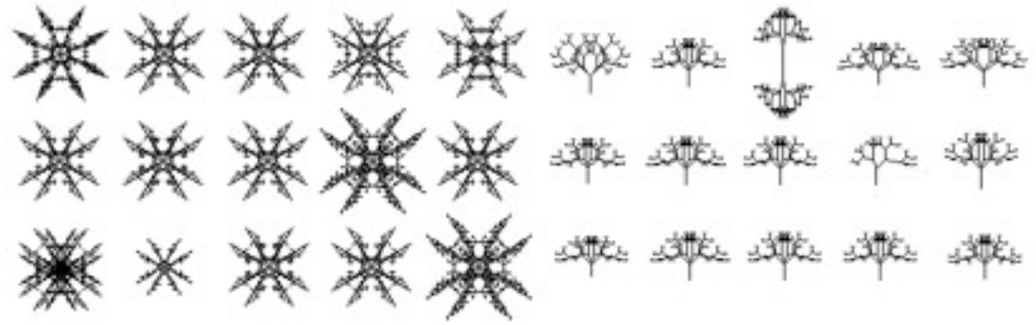
The principle was demonstrated neatly by zoologist Richard Dawkins in his interactive software, the *Blind Watchmaker* (1986). Use of the software begins with the computer presenting to the user a grid with a small population of automatically generated computer-drawn stick figures. These are effectively a tiny sample from the infinite library of possible stick figures the code can generate. The user visually assesses the members of this population, assigning them a subjective fitness judged purely on aesthetic merit. For instance on how insect-like, tree-like, face-like, ugly or intricate the line drawings appear to the user. The user then selects with a mouse-click the most desirable form.

From here, the computer removes the existing population from the screen and generates a new population of the same size. This population is composed entirely of the offspring from the selected parent. The new population will, in general, more closely meet the preferences of the user than did the previous generation since the offspring inherit, with some variation, the traits of the preferred parent. This evaluation–selection–reproduction sequence continues over many "generations" until the user has bred a visual design to their taste. That is, they have "found" a suitable design within the range of possibilities that the software is able to generate.

Dawkins' software implements the processes of:

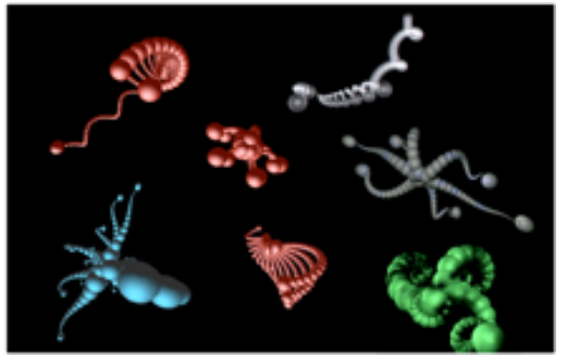
- Generating variation in a population.
- Reproduction with inheritance of the selected individuals.

The evaluation of fitness for individuals in the population is left up to the human user's imagination. Likewise, fitness-based selection of individuals from the population is performed by the human user manually selecting a satisfactory form from each new population.



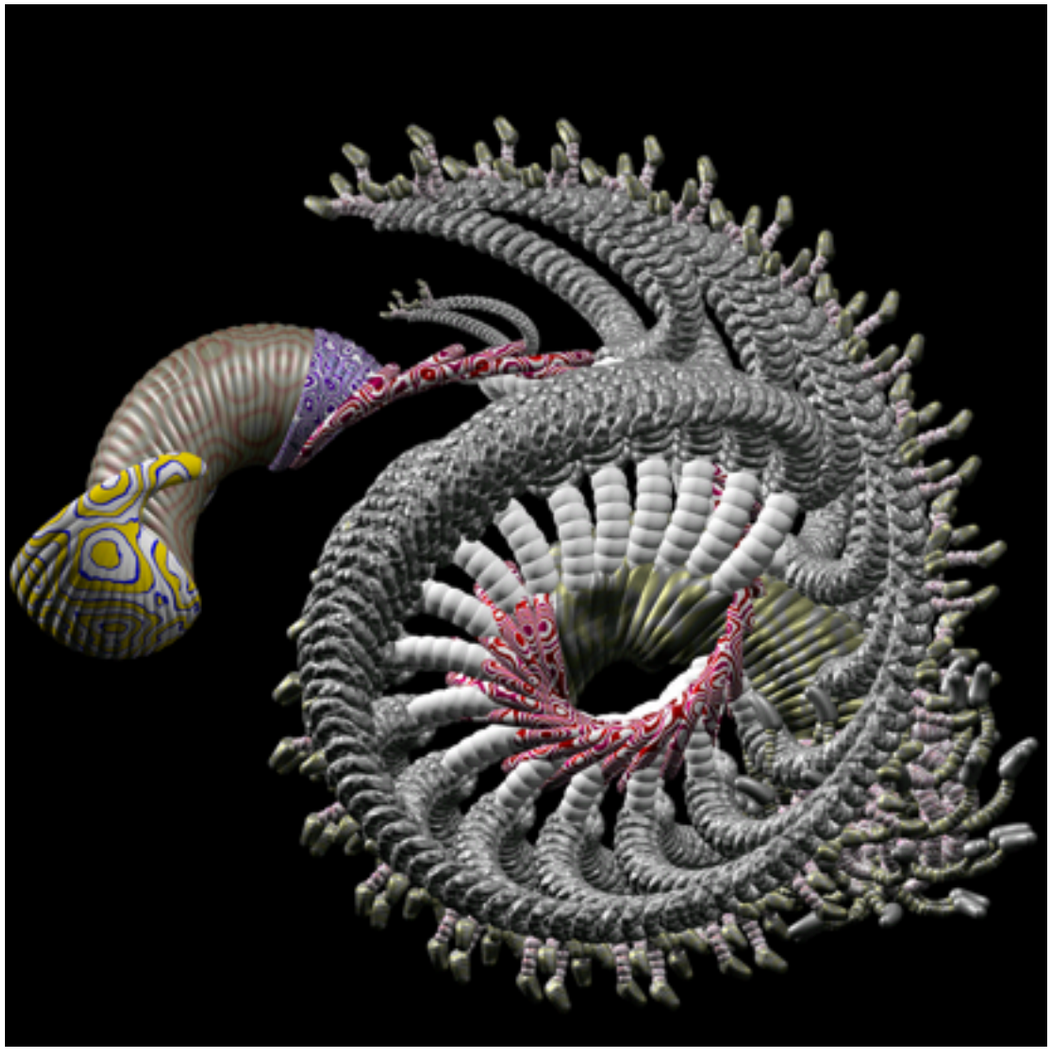
A population of structures bred over very few generations with Dawkins' *Blind Watchmaker* (1986). Dawkins created stick-figures that resembled trees, fish, insects, even characters from the English alphabet.

With some modification and variation this process has been widely used to make what is loosely referred to as "evolutionary art". It is possible to apply the technique to a wide variety of design tasks where human preferences are the best gauge of fitness. This is often the case for visual art and design, music composition and other subjective ventures. When the aim of the exercise is to derive a virtual creature's locomotory strategy or build a complete virtual ecosystem with evolving spe-



It is possible to breed 3D forms using aesthetic selection acting on virtual models. These forms were bred using the author's *Bauhaus* software, 1994-5.

cies, manual interference is often undesirable and usually quite impractical. To eliminate the need for a human in the loop the computer requires an explicit means to measure the fitness of members of a population.

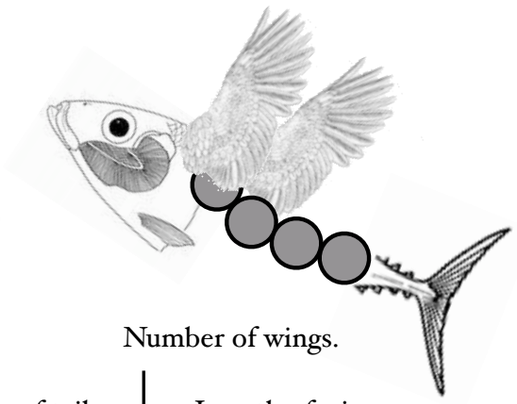


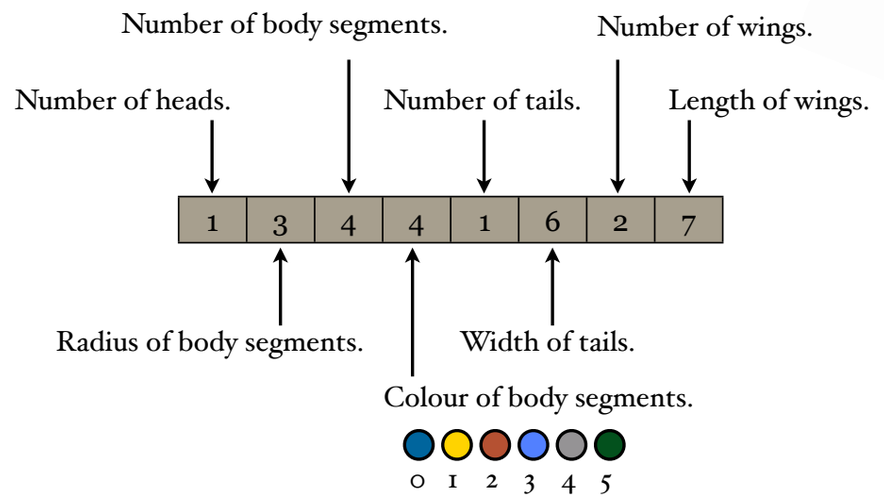
British artist William Latham has been aesthetically evolving virtual sculptures and computer animations since the 1980s. His software *Mutator* was written with Stephen Todd. Image © William Latham 1992. Used with permission.

Genotypes, phenotypes and fitness functions

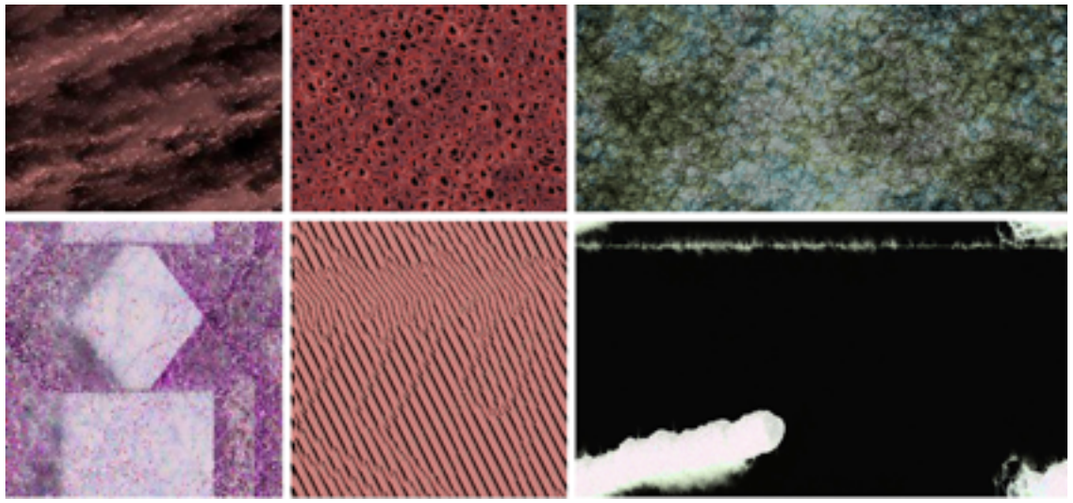
On earth, every cell includes a DNA molecule that, within the right environment, acts in accordance with natural chemical and physical laws to generate an organism. The organism may be only that single cell with a very simple behavioural repertoire, or it may be a multicellular primate capable of science, philosophy, art and writing ebooks. The DNA in all these cases is referred to as the *genotype* for the organism. The organism and its behaviour constitute a *phenotype*.

The genotype of a phenotype is passed down to offspring during reproduction. Through this medium offspring inherit phenotypic traits such as the morphological, physiological and behavioural characteristics of their parents. Natural or human selection then acts on the pheno-





As can be seen from this strange flying fish, a digital genotype can encode just about anything!



Textures generated by aesthetic selection of virtual ecosystems of drawing agents using *EvoEco*. © Kowaliw, Dorin, McCormack, 2011.

typic traits, governing which phenotypes can pass on genotypes to offspring.

To implement the evolutionary process in software a genotype must be coded. This can be as simple as an array of bits, an approach used in *Genetic Algorithms*. These are a type of evolutionary algorithm promoted by American complex adaptive systems researcher John Holland. A single or multi-dimensional set of integer or floating point numbers also makes a suitable genotype. These values could code for the presence or absence of phenotypic traits, or as line lengths, positions and orientations in a phenotype that is a stick-figure drawing. The numbers might be parameters for the control of the limbs of a virtual creature being evolved to walk, or they could specify environmental conditions under which a creature attempts a high-level behaviour such as fleeing, eating or mating.

Could you write a controller to dynamically balance these creatures on their feet? Could you extend it so that these creatures can walk, run, hop and jump over obstacles? Evolutionary processes can do (and have done) exactly that! Image © Ben Porter 2009. Used with permission.

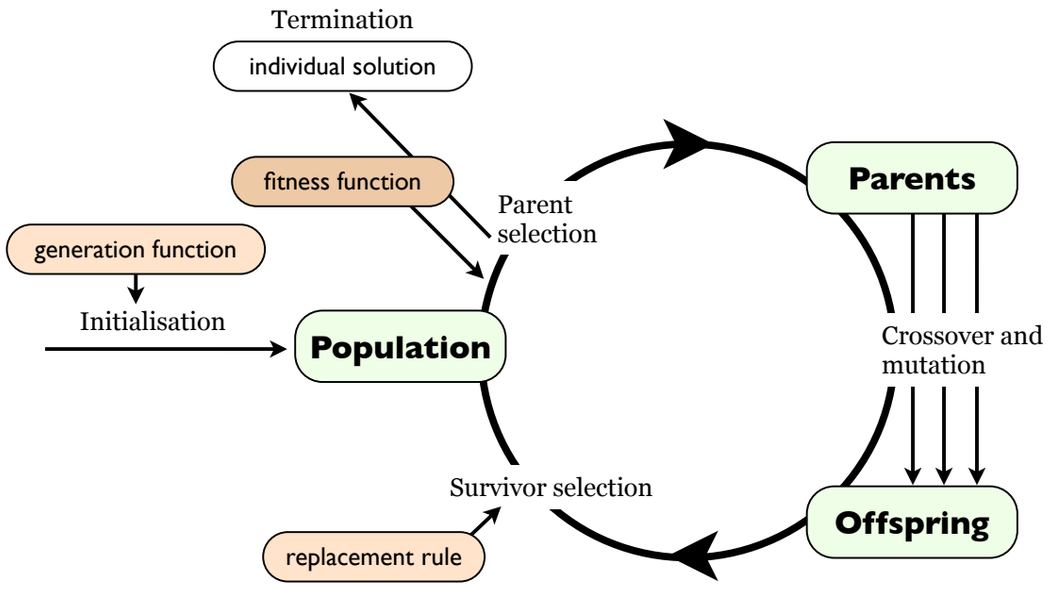
An alternative genotype structure is a tree. This idea has been promoted by the computer scientist John Koza and forms a part of *Genetic Programming*, another sub-field of the general area known as *Evolutionary Computation*. Tree genotypes can encode phenotypes that are mathematical expressions, complete computer programs, or algorithms for constructing phenotypes that are virtual organisms with hierarchical body-plans. Cyclic graphs can also be suitable genotypes for some problems.

Once a genotype and its mapping to a phenotype have been devised, an explicit *fitness function* must be written. This function scores each phenotype in a population so that the more closely it meets the desired goal, the higher its fitness. A suitable fitness function might be the distance a virtual creature stumbles from its starting position in a fixed time. Fitness functions might measure the efficiency of engines, or their top speeds, or they might measure a balance of power, reliability and efficiency. Fitness functions must permit the explicit measurement and comparison of a variety of phenotypes. The more finely they are able to distinguish between phenotypes with different capabilities, the more successful they are likely to be in defining a smooth fitness landscape.

Automated evolutionary computation

Now that the basic concepts of the fitness landscape, genotype, phenotype relationship and fitness function have been explained, it is possible to detail the evolutionary algorithm.

The algorithm cycles clockwise around the loop illustrated here, beginning with initialisation of the population.

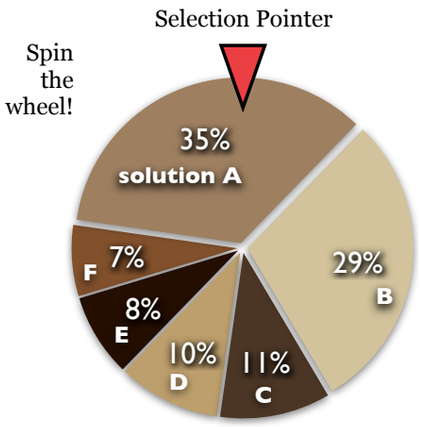


1. Initialisation. This is the creation of the first population of individual phenotypes. The initialisation of a population of phenotypes is typically carried out by randomly creating the sequences of numbers that represent the genotypes from which they are derived. Although suitable population sizes vary depending on the problem to be solved by the algorithm, from 50 to 200 individuals is common where an explicit fitness function will be applied.

Once the genotypes have been randomly created, phenotypes are built from these in a problem-specific way. For example the numbers might be used to specify a 3D model of a creature and a controller for its walk cycle.

2. Fitness evaluation. The next step is to test the phenotypes and measure their fitness. One approach is to choose pairs of phenotypes and play them off against one another in a game. Another approach is to let each phenotype do its own thing unencumbered for awhile, for example to see how far it walks. The fitness function takes the performance of each phenotype and turns it into a number. If any phenotype gets a perfect score (for any real problem this is unlikely in the first generation) then the algorithm
can stop here – the problem has been solved! Otherwise, the fitness scores are used for the next phase of the algorithm.

3. Parent selection. For evolution to work, the phenotypes with the highest fitness must be most likely to reproduce. The probability a phenotype will be chosen as a parent is sometimes computed using a "roulette wheel". The fraction of the wheel a phenotype occupies

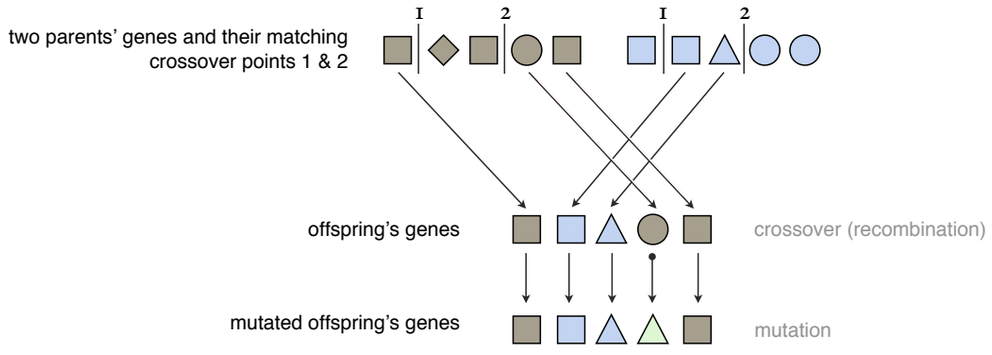


Different methods for using the fitnesses of solutions to select parents exist. One simple method is called *roulette wheel selection* using a *fitness-proportionate* or a *ranked* system to select the wedge sizes.

is governed by its fitness relative to the total fitness of the whole population. Other approaches are also used in different situations.

4. Reproduction (crossover and mutation). Once parents have been selected the mechanism of *crossover* is used to take a fraction of the genotype from each to copy into the offspring's genotype. Crossover shuffles or recombines the existing genes in a population among offspring.

Gene copying is inexact, and very occasionally, a random mutation occurs. Consequently, mutation is a process that may introduce completely new gene values to a population.



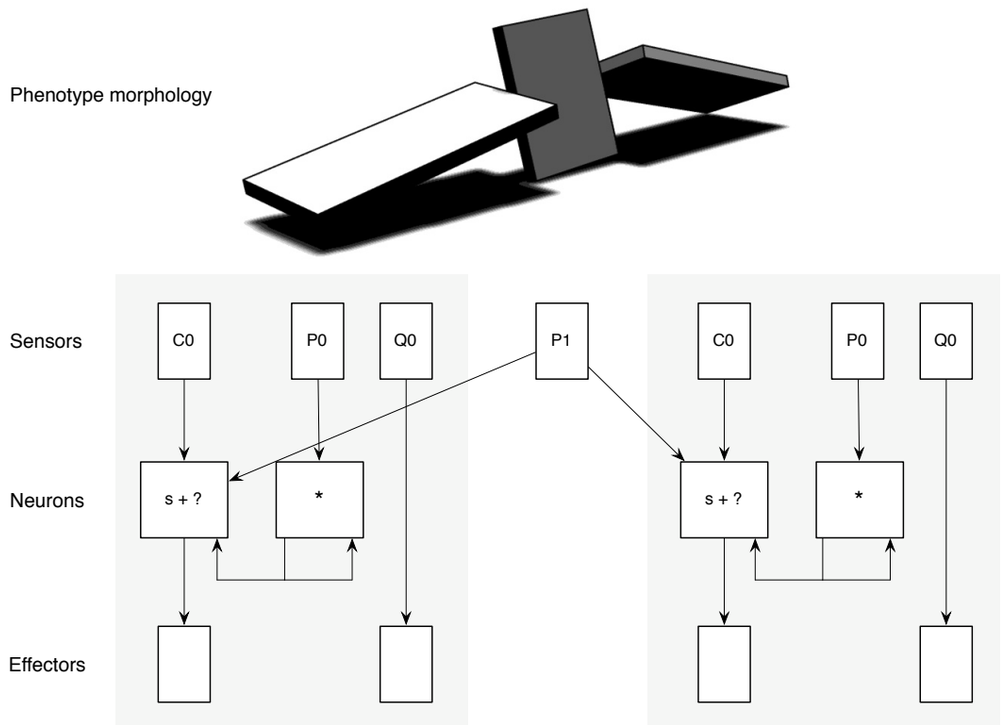
In the figure, two parent solutions are mated together. The operations that occur to produce a child are *crossover* (recombination) and *mutation*. Crossover requires one or more locations within the genotype to be selected randomly. The location of these *crossover points* determines which genes from each parent are spliced together to appear in their offspring. Occasionally, mutation randomly varies one of the child's genes or discards it and generates a new one from scratch.

5. Survivor selection. For each generation of the evolutionary loop a *replacement rule* decides which members of the existing population will be replaced by new offspring. In some cases the entire population is destroyed and replaced with new offspring after every generation. It is also possible to pick members of a population for removal by repeatedly selecting one at random and comparing its fitness against that of a newly generated phenotype. The fitter of the two is granted a space in the new population. The overall population size is usually kept constant.

The evolution loop is executed for multiple generations and, if everything runs to plan, the mean fitness of the population gradually increases until a perfect (or workable) solution is identified by the fitness function. The number of generations needed before a solution is found depends on many things. For instance it depends on the difficulty of the problem, how well the genotype was devised, the population size, the parameters for crossover and mutation and the ability of the fitness function to distinguish phenotypes. Trial and error helps to get things working. An evolutionary run that cycles through a few hundred generations is not unusual. By monitoring a fitness versus time graph it is possible to see if further generations are likely to lead to a solution, or if the fitness is failing to improve and a new run or some further re-jigging is necessary before starting afresh.

Artificial life and automatic evolution

One spectacularly successful application of evolutionary computation with explicit fitness functions was devised by American researcher and artist Karl Sims. His *Blocky Creatures* are made entirely, as their name suggests, of rigid blocks of different dimensions. Some creatures are built of only a few blocks, others are more complex and include several articulated components, often in repeated segments. The blocks of a creature are linked at joints. During a simulation in which creatures are placed into a virtual physical environment, the forces at the joints are controlled by a program that is evolved simultaneously with the body. The program and creature morphology both have genotypes with recursive graph structures. In this way their evolution runs in parallel and every body segment has the opportunity to evolve along with its own controller. A central processing component for coordinated control of body segments can evolve also.



A figure showing one of Sims' simpler creatures, and the corresponding "neural" structure that controls its behaviour via the application of forces to each of its joints. After Sims 1994.

Each creature's blocks can be equipped with sensors for detecting joint angles, contact, or the direction to a light source. The data from these sensors is processed by the evolved control program which is a graph of many interconnected nodes. The output of this network can apply forces across the joints of the creature's body blocks to move it about in response to its sensed environment.

By using explicit fitness functions, Sims evolved creatures capable of stumbling, rolling or walking across a simulated plain. He evolved swimming creatures, and creatures that jump into the air. Some creatures evolve to follow light sources. Using a co-evolutionary approach where two creatures were played off against one another for fitness, he was able to evolve morphologies and behaviours for games in which the virtual players competed for possession of a puck. The life-likeness of these abstract creatures makes them a landmark in digital evolutionary artificial life.

Further reading

Darwin, C. (1859), "On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life"

Dawkins, R. (1987). "The Blind Watchmaker". New York, W.W. Norton & Co.

Dorin, A. (2001). *Aesthetic Fitness and Artificial Evolution for the Selection of Imagery from the Mythical Infinite Library*. Advances In Artificial Life, 6th European Conference on Artificial Life. Kelemen and Sosik (eds). Prague, Springer-Verlag. LNAI2159: pp. 10-14.

Holland, J. H. (1975). "Adaptation in Natural and Artificial Systems, An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence", University of Michigan Press.

de Jong, K. A. (2006). "Evolutionary Computation, a unified approach", MIT Press.

Kowaliw, T., McCormack, J., Dorin, A., (2011). "An Interactive Electronic Art System Based on Artificial Ecosystemics", 2011 IEEE Symposium on Artificial Life, April 11-15, Paris, France, IEEE, pp. 162-169.

Koza, J. R. (1992). "Genetic Programming: on the programming of computers by means of natural selection", MIT Press.

Sims, K. (1994). “Evolving virtual creatures”, in Proceedings SIGGRAPH '94, Orlando, USA, ACM Press: pp. 15-22.

Todd, S. and W. Latham (1992). "Evolutionary Art and Computers". San Diego, Academic Press.

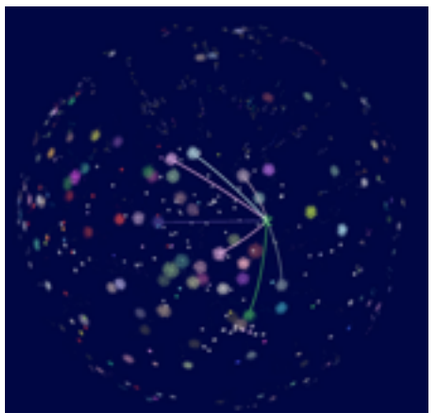
Wallace, A.R. (1858), “On the Tendency of Varieties to Depart Indefinitely From the Original Type”, Journal of the Proceedings of the Linnean Society of London, Zoology 3: pp. 46-50.

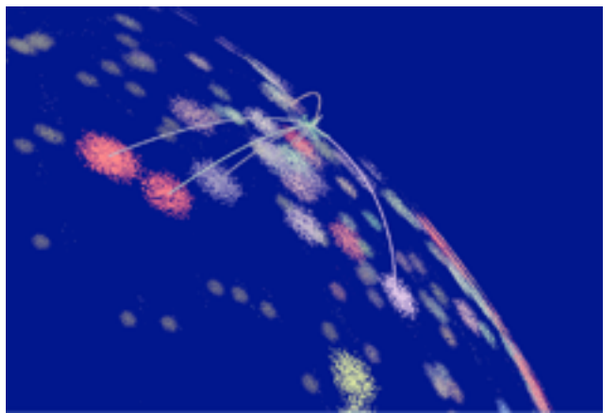
SECTION 3

Evolving ecosystems

TOPICS

1. PolyWorld
2. Tierra





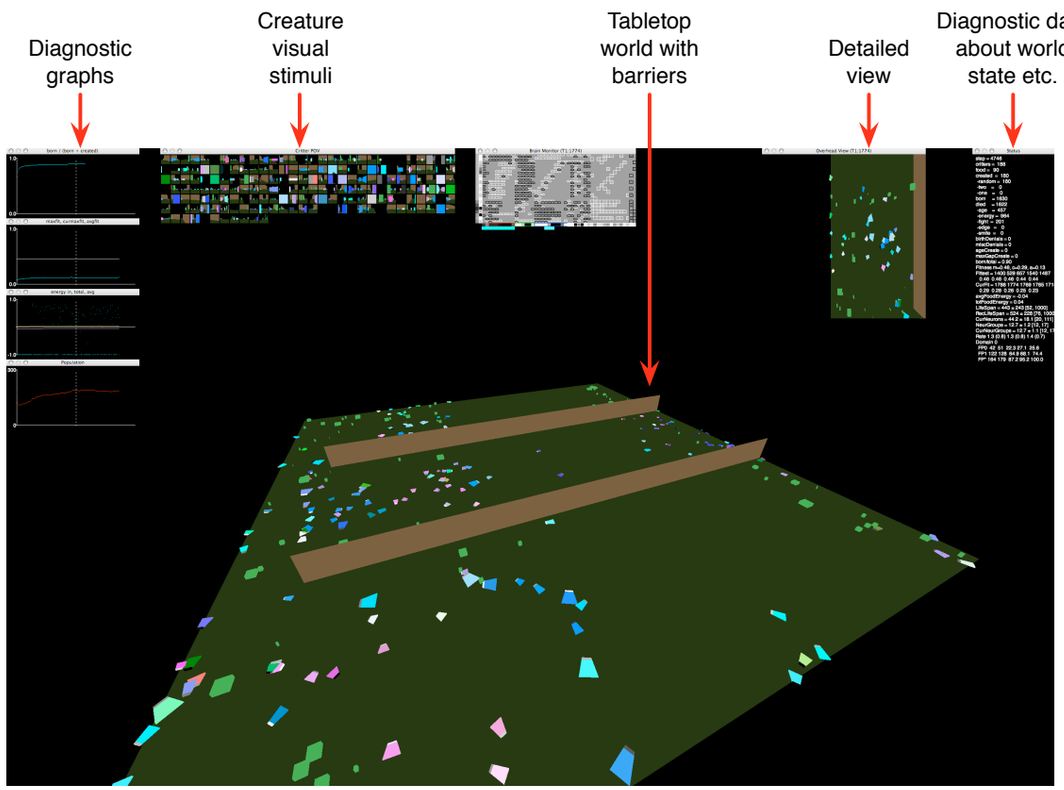
An evolving ecosystem, *Listening Sky*, of puff-ball agents roams a virtual globe searching for mates and playing musical patterns. Agents prefer mates that have bodies in their favourite colours and that sound pleasing to their personal musical tastes. A human controls a "listener" that hovers above the globe, eaves-dropping on the local populations of agents, exploring the dynamic soundscape that evolves.

Listening Sky, interactive VR, generative software installation. A collaboration between Rodney Berry, Wasinee Rungsarityotin and Alan Dorin. ATR, Kyoto, Japan, 2001.

Evolution generates and drives ecosystems. If we want to understand long-term ecosystem behaviour through simulations, we mustn't omit the evolutionary process. Not only is evolution essential in addressing questions of relevance to ecology, evolutionary computation can also play a role as a search algorithm. As a search algorithm it can assist us to find model parameters that create a match between virtual behaviour and data gathered from the real world. In this section though, we will focus on the former application. We will discuss a few evolving virtual ecosystems in detail to see how they work and how they can be used as research tools. Of great interest (not only to this author) is the potential for these systems to generate complexity of organism morphology and behaviour. It *seems* like nature is capable of evolving greater and greater complexity, however problematic it might be to measure this. But so far, our digital evolutionary processes have failed to live up to expectations. Why might this be? Could artificial ecosystems ever overcome the limitations built in to standard evolutionary algorithms in this regard?

PolyWorld

While working at Apple Computer in the early 1990s, Larry Yaeger developed *PolyWorld,* a virtual ecosystem that incorporated evolving agents with neural network brains. His aim was to build a system complex enough to be used for studying interactions of species in real ecosystems. PolyWorld wasn't the first virtual ecosystem model, but it incorporates many interesting features and so serves here to outline construction principles and the significant potential of the idea.



Larry Yaeger's *PolyWorld*. Image © Larry Yaeger 1992. Used with permission.

Space. PolyWorld is a tabletop that may be set up as a torus, or with dangerous edges over which its inhabitants tumble to their deaths. The space may contain internal barriers designed to restrict mixing between creatures in different areas, or possibly to act as cover for prey and predators.

Behaviour. The agents in PolyWorld appear as coloured polygons roaming the tabletop. During the simulation, at any stage they may execute the following behaviours determined by the output of their neural network brains.

Eat – Replenish energy resources. Condition: agent's location overlaps food *and* its eat behaviour is triggered.

Control field of view – Control the horizontal cone of vision.

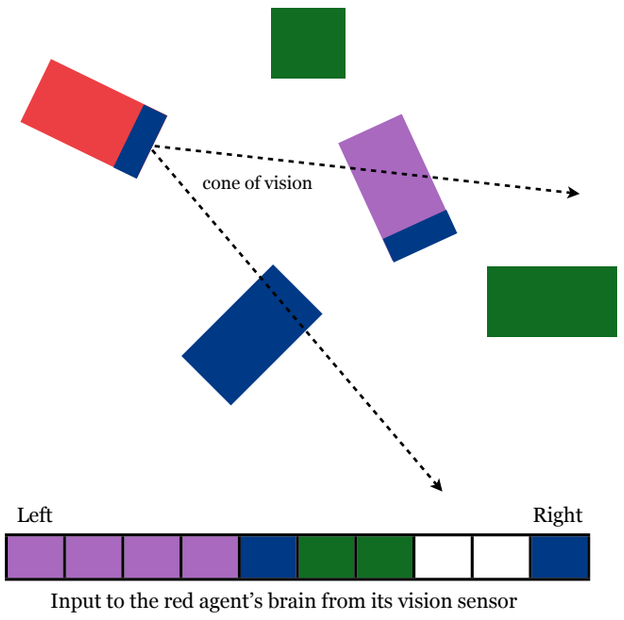
Change colour brightness – Control the brightness of polygons on its front.

Mate – Reproduce. Condition: agent's location overlaps another's and *both* organisms' mating behaviours are triggered. This is visually represented on an agent's body as blue colouration.

Move – Step forward an amount.

Turn – Turn an amount.

Fight – Attack another organism. Condition: agents overlap and *one* agent's fighting behaviour is activated. This is visually represented on the agent's body as red colouration.



An agent looks out into its environment. The inputs to its cognitive system are represented as a 1D array of coloured pixels extracted from its cone of vision.

Energy and matter. Agents metabolise energy for all activity, including thought and while sitting still. Energy is acquired by executing the *eat* behaviour near food. Food is visualised as green polygons scattered around the tabletop for which agents may search. This grass-like stuff grows freely at a rate and energy value that may be controlled by the programmer. Its location

is determined stochastically.

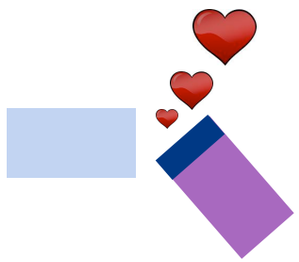
Dead organisms also offer energy to those who would eat them. An agent dies when its *health* energy reaches zero, but a dead organism may still have a non-zero food energy value. This makes predation a possibility within PolyWorld. It is important to note that although agents have a size and this effects fight outcomes and other activities, when they eat one another, matter/mass is not exchanged.

Sensation. PolyWorld agents see their local surroundings as a one-dimensional array of coloured pixels spanning the horizon in front of them. They may control the arc that their visual system subtends so as to be able to focus attention or take in a wide vista. The colours their visual system detects are fed into their neural network brains, along with a value representing their current health (so as to be able to know if they are being attacked for instance) and a random value. This latter parameter helps to promote generalisation in the neural network as well acting as a source of noise like that encountered by any real sensor.

Thought. PolyWorld agents are controlled by neural networks that process the inputs of the sensory system and convert them into instructions to execute behaviours. Each output of the neural network corresponds to one of the agent's basic behaviours. In some cases if an output neuron fires, the agent attempts to execute the corresponding behaviour (e.g. eating, mating). In some cases, the strength of the firing output indicates the degree to which a behaviour is executed (e.g. the distance moved or turned).

Agent neural states are updated synchronously. Each agent employs a learning algorithm to adjust its configuration after every time step. In this way agent responses are not fixed for their lifetime. PolyWorld creatures are capable of learning associations between what they perceive, what they do, and the state of their environment.

Genes and sex. PolyWorld agents each have a set of 8-bit genes. A number of these determine physiological traits which then influence agent behaviour. For example a gene encodes creature size and also effects its metabolic rate, fighting capability and the maximum amount of energy it can store. Other physiological genes include strength, maximum speed, life span and the fraction of energy that will be donated to offspring. Some genes also code for reproductive parameters specifying the mutation rate and the number of crossover points used.



Is it love, lust or fate that drives PolyWorld agents together?

Each agent's artificial neural network brain is also constructed according to values available for evolution. Genes specify the numbers of neurons devoted to red, blue and green vision components; the number of internal neural groups and their composition of inhibitor and exciter neurons. Parameters for connections between neurons and the learning algorithm are also encoded in genes.

As with any digital evolutionary problem worth solving, when PolyWorld starts with a population of randomly generated genotypes and derives phenotypes from them, the creatures won't be very successful. After all, they are going to be driven by untrained un-evolved artificial neural networks! They won't eat when they are hungry, they won't move when they are attacked, they may instead try to mate when they encounter a patch of grass, or they may do nothing at all.

To bootstrap PolyWorld, Yaeger implemented a genetic algorithm mode in which the fitness of agents is measured at predetermined times according to a predetermined fitness function. The fittest organisms are selected and mated together using a standard evolutionary algorithm. During the generations of this period, it doesn't matter if the parents are adjacent, nor even if they "want" to mate. Once the agents are able to manage to sustain themselves independently and find mates on their own, PolyWorld enters free running mode. Then everybody selects mates and reproduces under their neural networks' control. In this mode agents wishing to mate must both fire their "mate" output (rape is not possible here) and they can donate a gene-determined fraction of their energy to offspring to help get them going.

PolyWorld ecology. The reason for implementing PolyWorld in the first place was to see if interesting agent "species" and behaviours would emerge within this virtual ecosystem. Since a suite of basic behaviours was hardcoded to begin with, it wasn't so much a question of whether or not the ability to walk or change colour might appear, but perhaps the hard-coded behaviours could be coupled together by evolution in agents. They might then practice higher level composite behaviours that were *not* hard-coded, or even expected.

Yaeger reported the emergence of several types of agent. The simplest he dubbed "frenetic joggers" for their tendency to run straight ahead at full speed (in a torus world since otherwise they would have fallen off the world edge). These agents always wanted to mate therefore any others of their kind that they encountered offered reproductive opportunities. Similarly, the agents' eat neurons were always firing so any food they passed was a potential source of nourishment. A variant of these creatures, the "edge runners" travelled around a bounded world with a similar approach to finding mates and energy. The "indolent cannibals", occupied virtual cities. These were sedentary communities where everybody mated with their neighbours and ate them when they died.

Interesting couplings between hard-coded behaviours also emerged. For example, some creatures sped up in response to visual stimuli, some ran away when attacked or actively fought back. Some creatures slowed down to graze when they encountered food or even sought it out and circled it. Some agents were even seen to have developed the ability to follow others.

Summary. PolyWorld is an agent-based ecosystem model incorporating models of several significant features of real biology, especially with regard to learning and evolution. While it doesn't realise the complexity of natural ecosystems, nor does

it seem to show the open-ended expansion of complexity apparent in biological evolution, PolyWorld suggested the possibilities that virtual ecosystems might hold for research into real ecosystem dynamics.

A number of potentially vital aspects of real ecosystems are missing if PolyWorld is to exhibit the apparent open-endedness of biological evolution, we will explore some of these in detail later. One feature that might be worth examining is the possibility for PolyWorld organisms to live on and within one another's bodies. Even if this was the only extra feature implemented, it would enhance the opportunities for parasites and symbionts to evolve. In the system described next, PolyWorld-like predation cannot appear, yet symbiosis and parasitism are easily evolved.

Tierra

To understand the dynamics of evolutionary processes and species' interactions, a literal model of biological ecosystems is not strictly necessary. The ecologist Tom Ray adopted an abstract computational view of virtual ecosystems when he developed *Tierra* in the early 1990s.

Tierra does not *look* like any kind of ecosystem we would find in nature, it lacks a ground plane and has no visible creatures walking, eating, mating or fighting for instance. But its underlying dynamics are based on the evolution of agents competing for space and resources all the same.

Space, energy and matter. Tierran space is the finite memory of a *virtual machine* that can execute virtual assembler language programs using its own operating system. The memory takes the topological form of a 1D array with its ends connected to form a loop of cells.

The instructions in Tierra's virtual assembler language are loosely analogous to matter. Instructions can be written into and read from cells in Tierran space, and assembled into working programs that constitute agents, *Tierrans*. Agents compete against one another for space in the virtual machine memory. They use slices of virtual central processing unit (CPU) time to execute their instructions via the virtual operating system. Hence, Tierra's CPU time is analogous to energy.

Behaviour. Tierrans are programs built from assembler code sequences. Thirty-two different instructions exist in the machine language (e.g. MOV, CALL, RET, POP, PUSH, COPY); basic instructions that will be familiar to any programmer. Addressing is by template. So an instruction to JMP (jump) to a location in memory is not specified by a numeric address (there are no numeric operands), but instead by a template. Under program control execution may jump to the nearest memory location, in either direction, which matches the template.

Each Tierran is allocated exclusive *write* permission to a block of memory. However, it may *read* or *execute* any memory cell in the virtual machine, even if it contains instructions within

the block allocated to another agent. In this way a program may use code that is not a part of its own body.

Tierrans are positioned on a circular "slicer" queue controlled by the virtual operating system. This governs the timing with respect to other agents, and the amount of time, that each program can spend executing its instructions on its virtual CPU. The amount of CPU time allocated to a Tierran can, if desired, be tied to the program's length. In this way it is possible to favour different sized programs. For instance, if the slice length is kept constant for all Tierrans, the environment naturally favours Tierrans able to act quickly.

Genes and sex. As already explained, each organism has exclusive *write* access to its own block of memory into which it can place virtual assembler code. The list of these instructions appearing in any one block constitutes the genotype of that Tierran. Its phenotype is the behaviour that the genotype produces when it is executed on its virtual machine CPU in its environment.

A Tierran can request write access to an extra memory block. This is achieved with the MAL (memory allocation) instruction. A parent can then write code into the allocated daughter cell to grow or replicate. Replication occurs if the program copies itself into its daughter cell and executes a DIVIDE instruction. This separates the daughter from the parent memory block so that it gains exclusive write access to itself and will be allocated CPU time via the slicer queue like any other Tierran. After division, the parent can request the allocation of a new daughter cell.

To bootstrap *Tierra*, Ray filled the memory with a mixed primordial "soup" of 60,000 instructions. Importantly, he also hand-coded a replicating Tierran ancestor. This program locates templates that mark its beginning and end in memory so that it can calculate its own size. It then requests a daughter cell of the right size, copies itself into this, and divides from it.

The Tierran replication process is imperfect. Sometimes bits are randomly flipped during instruction copying. This has an effect analogous to mutation in a standard genetic algorithm. Also, Tierran assembler instructions don't always work as planned. For instance, an instruction to flip a bit act on the wrong bit, or not do anything at all. Additionally, at some specified background rate the operating system randomly flips bits in the virtual memory. This is likened by Ray to cosmic radiation introducing permanent and heritable mutations into living Tierrans. All of this noise helps to scramble events in Tierra so that, as might be imagined, the memory rapidly fills with imperfect copies of the ancestral replicator, mutants! What happens when the memory starts to fill?

Limits to growth. A "reaper queue" is maintained in *Tierra*. Organisms that reach the head of the queue are culled. Their instructions are left as dead matter in the memory but aren't allocated CPU time by the slicer. Newly freed memory is allocated as daughter space for holding new arrivals.

New Tierrans begin at the tail of the reaper queue. They move away from its head when they successfully execute an instruction (for instance they request a jump to a template and that template is found in the memory); and move towards the head of the reaper queue when an instruction is unsuccessful. Active and successful organisms may prolong their time in *Tierra* a little. But the older a Tierran gets the closer it gets to the abattoir.

Tierran ecology. Once things get going in *Tierra*, a diversity of programs appears. This isn't quite the open-ended complexity increase Ray sought, but it is a interesting nonetheless. Of particular interest to Ray was the appearance of various forms of parasitism. Tierrans evolve that seize control of the copy code of other organisms to copy their own instructions. Without the presence of hosts, these parasites are unable to replicate. Although they don't harm the hosts directly, they certainly compete for space and CPU time.

Some Tierrans appear that are immune to parasitism, and then this immunity is circumvented by newly evolved parasites. Hyper-parasites also evolve that subvert a parasite's energy to assist them in their own replication. Some Tierrans evolve that can only replicate socially. These interactions depend on an aggregation of neighbours (in space) catching and passing on one another's Jump-Template instructions.

It is impossible for *Tierra* to evolve predator / prey relationships since each agent maintains exclusive access to its memory cells. Similarly limiting, Ray notes, is the simplicity with which new instructions (matter) can be created, transformed and destroyed in *Tierra*. Any instruction can be written like any other, without regard to what was in a cell before. Instructions are not conserved in a way analogous to physical matter. Perhaps implementations of Tierra-like systems where these issues are accounted for would be more likely to result in the kinds of behaviour typical of real ecosystems: in biology, metabolisms evolve as a direct result of pressure to transform real matter to build bodies and extract energy from an organism's surroundings.

Core Wars was developed in the 1980s by Alexander Dewdney, a Canadian mathematician and computer scientist. Dewdney's software provides an environment in which hand-written computer programs do battle with one another in computer memory. Although the programs could potentially do anything they liked with their allocated CPU time, the battle between them had each program in the linear memory space attempting to destroy the others by overwriting or altering their instructions and attempting to repair any damage it sustained.

Summary. Tierra is an elegant and interesting approach to generating ecosystem-like behaviour within a very abstract environment. The idea that life might be a property identified with computation is, I feel, aesthetically suggested by such software. Others have adopted the idea of program-based environments to study similar questions. For instance, Andrew Pargellis used a system, *Amoeba,* derived from Tierra, to facilitate the spontaneous appearance of replicators in a random soup of machine code instructions. From this he explored the emergence of non-random

ecosystem-like dynamics among the evolving replicators and their parasites. Unlike *Tierra*'s looped linear memory array (which was adopted from a still earlier system called *Core Wars*), *Amoeba* employs a 2D grid of virtual CPUs to define neighbourhood relations between programs. Amoeba shares this feature with Chris Adami and Titus Brown's *Avida* system, another platform dervived from *Tierra*. *Avida* continues to support research into evolutionary systems.

Further reading

Yaeger, L. S. (1992). *Computational Genetics, Physiology, Metabolism, Neural Systems, Learning, Vision and Behavior or Polyworld: Life in a New Context*. Artificial Life III. C. Langton, Addison-Wesley: 263-298.

Ray, T. S. (1990). *An approach to the synthesis of life*. Artificial Life II. C. G. Langton, C. Taylor, J. D. Farmer and S. Rasmussen. Santa Fe, New Mexico, Addison Wesley. X: 371-408.

Pargellis, A. N. (2001). "Digital Life Behavior in the Amoeba World." Artificial Life 7(1): 63-75.

Dewdney, A. K. (1984). In the game called Core War hostile programs engage in a battle of bits. Scientific American, 250, 14–22.

Adami, C. and C. T. Brown (1994). "Evolutionary Learning in the 2D Artificial Life System 'Avida'". Artificial Life IV, MIT Press, pp. 377-381.

Assembler code for a Tierran ancestral replicator

```
nop1; 010 110 01 0 beginning marker
nop1; 010 110 01 1 beginning marker
nop1; 010 110 01 2 beginning marker
nop1; 010 110 01 3 beginning marker
zero; 010 110 04 4 put zero in cx
not0; 010 110 02 5 put 1 in first bit of cx
shl; 010 110 03 6 shift left cx (cx = 2)
shl; 010 110 03 7 shift left cx (cx = 4)
movcd; 010 110 18 8 move cx to dx (dx = 4)
adrb; 010 110 1c 9 get (backward) address of beginning marker -> ax
nop0; 010 100 00 10 complement to beginning marker
nop0; 010 100 00 11 complement to beginning marker
nop0; 010 100 00 12 complement to beginning marker
nop0; 010 100 00 13 complement to beginning marker
sub_ac; 010 110 07 14 subtract cx from ax, result in ax
movab; 010 110 19 15 move ax to bx, bx = start address of mother
adrf; 010 110 1d 16 get (forward) address of end marker -> ax
nop0; 010 100 00 17 complement to end marker
nop0; 010 100 00 18 complement to end marker
nop0; 010 100 00 19 complement to end marker
nop1; 010 100 01 20 complement to end marker
inc_a; 010 110 08 21 increment ax, to include dummy instruction at end
sub_ab; 010 110 06 22 subtract bx from ax to get size, result in cx
nop1; 010 110 01 23 reproduction loop marker
nop1; 010 110 01 24 reproduction loop marker
nop0; 010 110 00 25 reproduction loop marker
nop1; 010 110 01 26 reproduction loop marker
mal; 010 110 1e 27 allocate space (cx) for daughter, address to ax
call; 010 110 16 28 call template below (copy procedure)
nop0; 010 100 00 29 copy procedure complement
nop0; 010 100 00 30 copy procedure complement
nop1; 010 100 01 31 copy procedure complement
nop1; 010 100 01 32 copy procedure complement
divide; 010 110 1f 33 create independent daughter cell
jmp; 010 110 14 34 jump to template below (reproduction loop)
nop0; 010 100 00 35 reproduction loop complement
nop0; 010 100 00 36 reproduction loop complement
nop1; 010 100 01 37 reproduction loop complement
nop0; 010 100 00 38 reproduction loop complement
ifz; 010 000 05 39 dummy instruction to separate templates
nop1; 010 110 01 40 copy procedure template
nop1; 010 110 01 41 copy procedure template
nop0; 010 110 00 42 copy procedure template
nop0; 010 110 00 43 copy procedure template
pushax; 010 110 0c 44 push ax onto stack
pushbx; 010 110 0d 45 push bx onto stack
pushcx; 010 110 0e 46 push cx onto stack
```

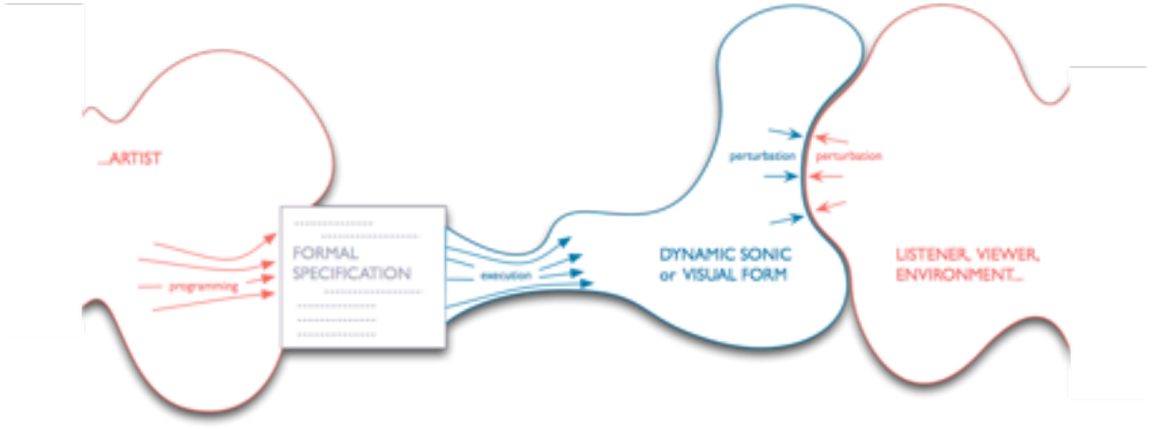
```
nop1; 010 110 01 47 copy loop template
nop0; 010 110 00 48 copy loop template
nop1; 010 110 01 49 copy loop template
nop0; 010 110 00 50 copy loop template
movii; 010 110 1a 51 move contents of [bx] to [ax] (copy one instr.)
dec_c; 010 110 0a 52 decrement cx (size)
ifz; 010 110 05 53 if cx == 0 perform next instruction, else skip it
jmp; 010 110 14 54 jump to template below (copy procedure exit)
nop0; 010 110 00 55 copy procedure exit complement
nop1; 010 110 01 56 copy procedure exit complement
nop0; 010 110 00 57 copy procedure exit complement
nop0; 010 110 00 58 copy procedure exit complement
inc_a; 010 110 08 59 increment ax (address in daughter to copy to)
inc_b; 010 110 09 60 increment bx (address in mother to copy from)
jmp; 010 110 14 61 bidirectional jump to template below (copy loop)
nop0; 010 100 00 62 copy loop complement
nop1; 010 100 01 63 copy loop complement
nop0; 010 100 00 64 copy loop complement
nop1; 010 100 01 65 copy loop complement
ifz; 010 000 05 66 this is a dummy instruction to separate templates
nop1; 010 110 01 67 copy procedure exit template
nop0; 010 110 00 68 copy procedure exit template
nop1; 010 110 01 69 copy procedure exit template
nop1; 010 110 01 70 copy procedure exit template
popcx; 010 110 12 71 pop cx off stack (size)
popbx; 010 110 11 72 pop bx off stack (start address of mother)
popax; 010 110 10 73 pop ax off stack (start address of daughter)
ret; 010 110 17 74 return from copy procedure
nop1; 010 100 01 75 end template
nop1; 010 100 01 76 end template
nop1; 010 100 01 77 end template
nop0; 010 100 00 78 end template
ifz; 010 000 05 79 dummy instruction to separate creature
```

SECTION 4

Aesthetic ecosystems

TOPICS

1. Meniscus (2003)
2. Constellation (2009)
3. Pandemic (2012)



Generative Art and its sub-category, Artificial Life Art are dependent on the specification by an artist of a dynamic process. This runs with some degree of autonomy from the artist and may actually constitute the artwork. Alternatively, the process may generate an outcome, such as an image, 3D model or text, that is considered the artwork.

Arguably, ecosystems count amongst our most valuable aesthetic experiences, providing us with exceedingly dynamic environments capable of engaging all of our senses. This emergent complexity is something many in Artificial Life strive for, it is also a major concern of artists engaged in Artificial Life art and generative art. Unsurprisingly given its history, Artificial Life is as interesting to artists as it is to scientists and philosophers.

Generative Art, and the subset of it concerned specifically with Artificial Life, is a practice in which artists establish processes that act more or less independently to generate, or even to *be*, artworks. Many generative and Artificial Life artists write computer software to realise their artistic processes. Programs offer great flexibility and potential to to meet creative aims. They can generate rich, dynamic audio-visual environments suitable for installation in a gallery, museum, or for display on cinema and virtual-reality theatre screens. Additionally, software can easily be connected to physical sensors that respond to temperature, humidity, movement, light and sound, or to data gleaned remotely via the internet. The diversity of potential perturbations to software via sensors creates opportunities for inter-

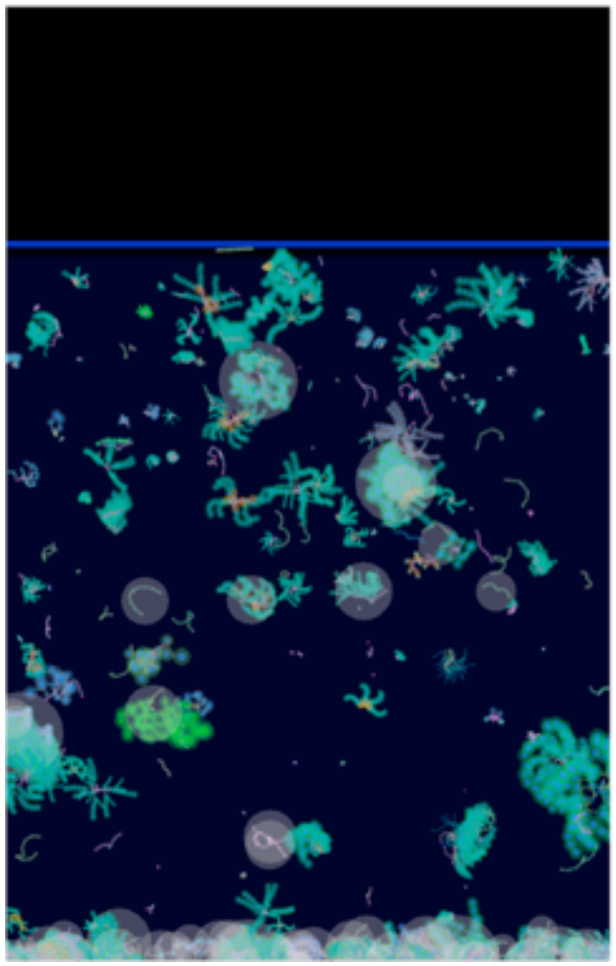
瘟疫 Plague (2006)

A virtual ecosystem audio-visual installation in which organisms and diseases co-evolve.

active Artificial Life Art, games, data visualisations and exploratory exhibits. Since the 1960s many artists have experimentally coupled sensors to robots and other hardware to create Cybernetic Art that explores situated feedback and autonomous control systems. This is fascinating and relevant to Artificial Life, but it falls just outside this book's focus on software. In the next subsections a few evolving ecosystems created by the author will be described. Each has its own aesthetic aims above and beyond the usual goals of Artificial Life research and ecosystem simulation.

> The VIDA Artificial Life art competition has been well documented and provides information on many superb examples of Artificial Life art.

I am not the only person to explore this aspect of Artificial Life art, but I hope the reader doesn't think it too self-indulgent of me to want to showcase a few of my own works! Here they are...



Meniscus

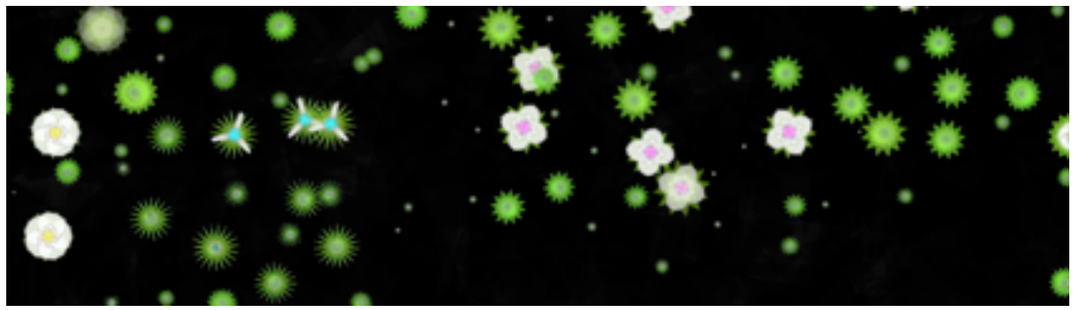
The diversity of organism structures and movement patterns found in nature is astonishing. Each is adapted through the evolutionary process to its specific niche. Can such diversity be realised in software? *Meniscus* is a virtual ecosystem filled with 2D virtual "microorganisms" that attempts this feat. The creatures flutter about in a sheet of water sandwiched between two glass plates; in reality a wall-mounted screen. The organisms have preferred water depths and levels of agitation at which they are most happy to search for and select mates. When a suitable couple is made, the two parents reproduce using crossover and mutation operations applied to their digital genome. Their offspring tumble to the bottom in an egg that jumps around for a genetically-determined amount of time until it is ready to hatch. From the egg emerges a creature resembling its parents. It moves towards its preferred depth in search of a mate, clustering among others of its kind, or seeking a space of its own depending on how comfortable it feels.

Human visitors to the gallery have control over the water level in the virtual ecosystem via a control dial. They may empty the water from the tank or fill it. They may repeatedly agitate the water by spinning the dial vigorously. These movements force the creatures into conditions they may find favourable or unfavourable

Meniscus (2003), an interactive Artificial Life artwork that employs indirect artificial evolution in a virtual ecosystem to generate complex organism morphology and movement.

depending on their genetically determined preferences. The creatures that prefer the conditions generated by the work's visitors come to dominate the population for awhile. But the system is dynamic. New visitors bring new conditions, even an absence of visitors potentially brings new conditions. The scene is never the same twice as new morphologies and creature movement patterns evolve over the weeks of an exhibition.

Constellation



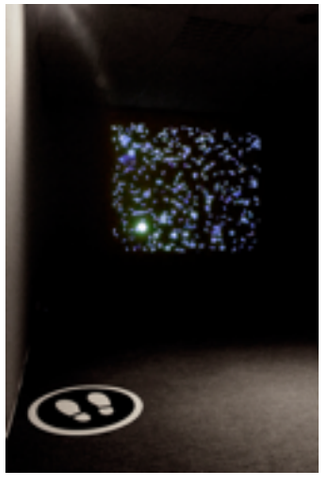
Constellation (2009) is an Artificial Life artwork in which a bed of flowers evolves over many hours. Each flower species is dependent on the environment for resources and in turn generates new resources that other species may come to depend on.

Organisms generate and occupy niches within ecosystems. Each species acquires resources from its environment, and each engineers new niches for other species to occupy by providing new raw materials, new physico-chemical conditions, new habitat and behavioural possibilities to be occupied by others. Behavioural niches are frequently explored in virtual ecosystems such as PolyWorld and Tierra. *Constellation* instead represents the construction of new chemical niches. The ecosystem is represented as a bed of tiny flowering plants occupying an initially uniform surface that provides some basic resources. A handful of randomly generated seeds is scattered and, if the resources are suitable, each grows into a small bundle of leaves before a flower buds. The resources each plant needs to survive are determined by its genes. As it extracts resources from its locality, each plant also generates some new resources in its vicinity that may be retrieved by other creatures.

The plants reproduce asexually when they reach maturity, scattering seeds sharing the parent genome (with some possible mutation) into their neighbourhood. If the seeds find the resources they need, they too flourish and generate new resources locally. Otherwise they die. The result is a dynamic patchwork carpet of flowers. Sometimes a plant evolves dependence on a resource that is only provided by another species during its lifetime. In cases like this the relationship results in the death of the dependent plants once their host fails due to old age or as part of a cascade of extinctions.

Pandemic

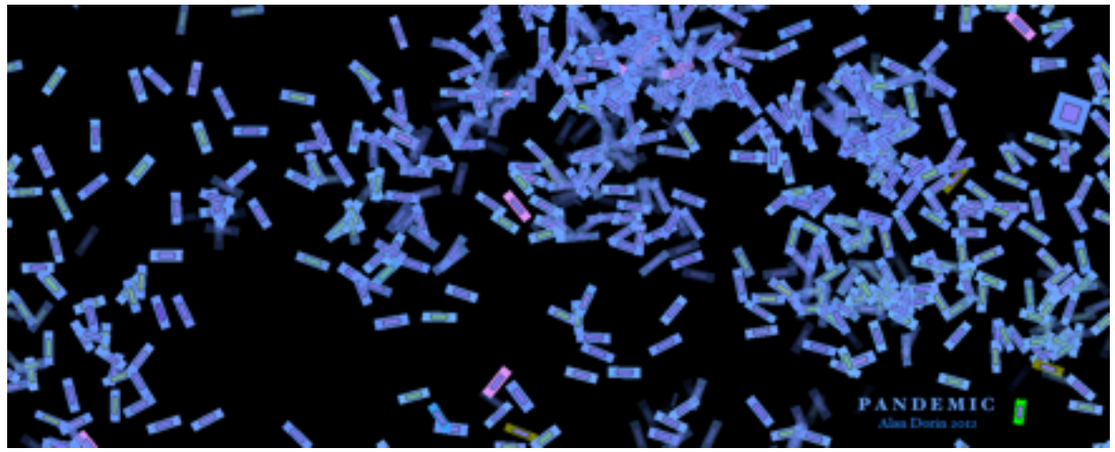
Pandemic is an individual-based virtual ecosystem in which organisms are represented as simple geometric structures. Each is specified by a digital DNA that codes the organism's colour, dimensions and behaviour. Each organism has preferences for mates of particular colours, sizes and aspect ratios. These it seeks out in a planar landscape. If two organisms encounter one another and are mutually attracted, they may reproduce. Their offspring inherits DNA from each of its par-



Any human watching *Pandemic* (2012) who steps within the marked circle will have their clothing colour read and projected into the virtual world within the disc (shown on the screened image in bright green). This acts as the point of introduction of new infectious diseases into the virtual world.

ents, and hence a mixture of their visual and behavioural characteristics.

The human visitor to the initial utopian Pandemic brings with them contaminants in the form of coloured clothing. The more colourful the garb the worse it will be for the virtual organisms since each new colour acts as a transmissible disease introduced by a video-camera grab of a visitor's clothing. If any virtual agent traverses the portal that links their world to the outside, they risk infection. From then on, as the organisms wander they may encounter another that is infected by a coloured disease. Perhaps this is carried by a potential mate, perhaps just by an organism that is passing by. If the colour of the disease to which they are exposed matches that of the organism, the infection spreads. Some organisms survive infection and later exhibit immunity when re-exposed to the same disease. Others perish. The process continues indefinitely as the organisms and diseases co-evolve in an ongoing arms race to out-compete one another and as new diseases are introduced by human visitors. The result is a vibrant and dynamic installation that is never the same on two occasions and to which visitors continue to bring new elements.



Pandémie / Pandemic (2012) is an interactive Artificial Life artwork in which human visitors inadvertently introduce new diseases to a virtual ecosystem. These infect, harangue, destroy and co-evolve with the virtual creatures.

Further reading

Dorin, A., (2004). "The Virtual Ecosystem as Generative Electronic Art", in Proceedings of 2nd European Workshop on Evolutionary Music and Art, EvoWorkshops 2004, Raidl *et al* (eds), Springer-Verlag, pp. 467-476.

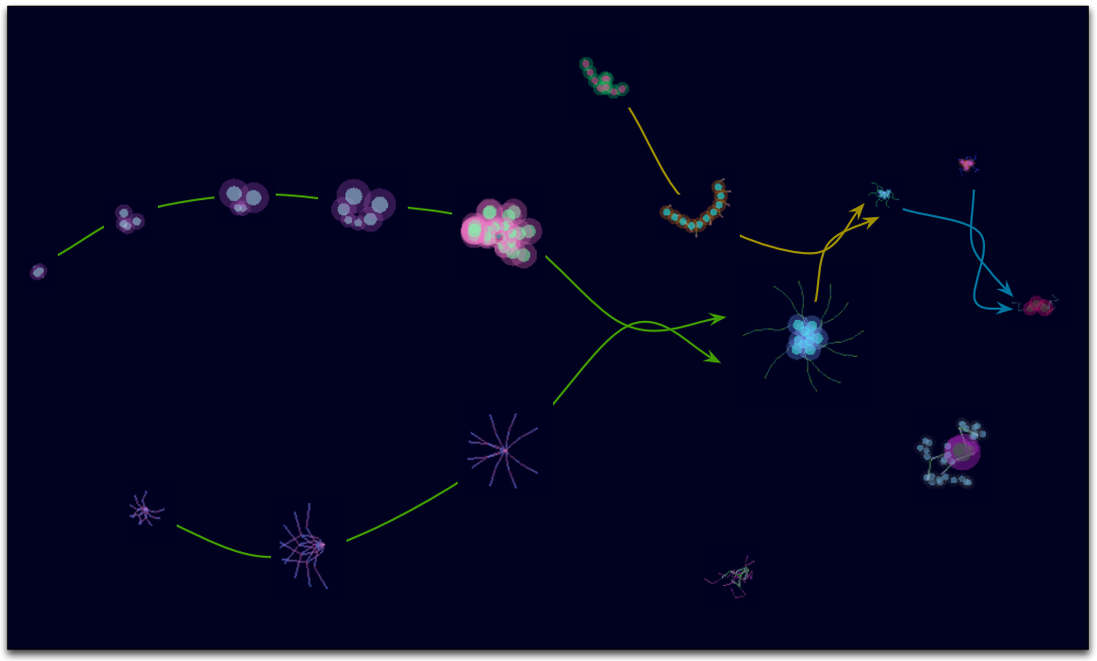
Dorin, A., (2008). "A Survey of Virtual Ecosystems in Generative Electronic Art", in The Art of Artificial Evolution, Machado & Romero (eds), Springer-Verlag pp. 289-309.

SECTION 5

Open-ended evolution

TOPICS

1. Is natural evolution open-ended?
2. A measure of organism complexity
3. Ecosystem complexity
4. Complexity increase in software



Some movement sequences and lineages produced by asexual/sexual hybrids in a virtual ecosystem, *Meniscus*, by the author.

Is natural evolution open-ended?

> *In a history of 3000 million years, evolution has not run backward.. . . Why is this? Why does evolution not run at random hither and thither in time? What is the screw that moves it forward, or at least, what is the ratchet that keeps it from slipping back? Is it possible to have such a mechanism which is not planned? What is the relation that ties evolution to the arrow of time, and makes it a barbed arrow?* – Bronowski, 1970.

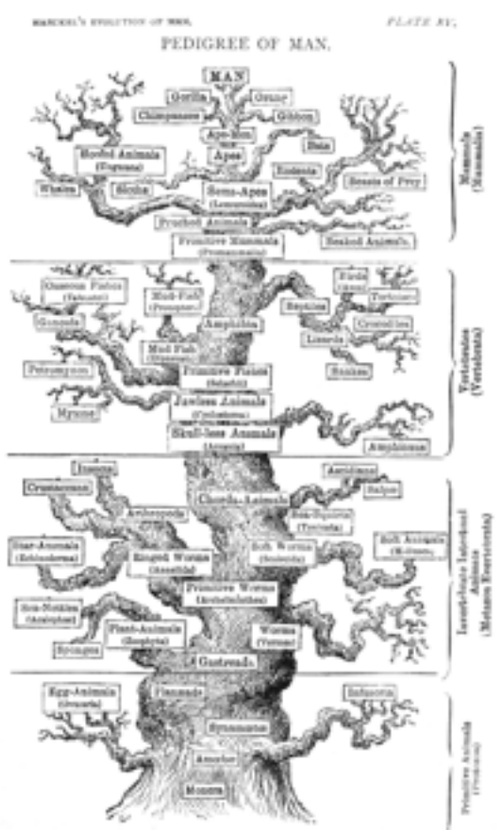
One hope of Artificial Life researchers using evolving virtual ecosystems as tools has been to explore ecological phenomena such as population dynamics, speciation, species ranges and resilience. Ecologists are using this approach too. But specifically within the field of Artificial Life, underlying much of the interest in ecological simulation is an agenda to realise computational "open-ended evolution". Researchers have wondered whether software can indefinitely generate new levels of organism complexity and new adaptive organism components. Biological evolution seems capable of both.

It appears that, yes, digital evolution can generate new adaptive organism components, but these are often not very interesting. They are trivial to generate and trivial to behold. Can new *and* interesting components be generated in software indefinitely? That is still undecided, partly because *interestingness* is so hard to define. Likewise, complexity increase, especially that which in nature appears hierarchically organised, has proven to be a tough nut to crack! This section explains why, and offers suggestions as to how to improve the situation.

To begin with we must address a concern of evolution theorists, *Does natural evolution generate increasing organism complexity?* Intuitively, it seems that prokaryotes are simpler than humans. It is also clear that the latter evolved for the first time much more recently than the former. Is it therefore true that biological evolution is *driving* an increase in the complexity of organisms? Or is the apparent complexity increase over time generated passively?

A measure of organism complexity

Before we can even begin to answer this question, we must agree on a definition and measure of organismic complexity. And right away our troubles begin since there is no consensus on an appropriate measure. Should we focus on the number of different cell types a species has? How do we rigourously distinguish between cell types anyway? Maybe instead we should be interested in a species' behavioural repertoire? Or morphological complexity? The information content of its DNA?



Note the position of capitalised MAN at the apex of the tree. This of course implies His superiority to all other lifeforms. E. Haeckel's image "Pedigree of Man" from, *The Evolution of Man: A popular exposition of the principal points of human ontogeny and phyologeny*, 1897, New York: D. Appleton and Company (p188, plate XV).

Even if we could agree on the detail of these methods, when making fine-grained distinctions between species, different measures generate different orderings. Some measures even topple humans from the apex of creation, a proposal that would once have been considered sacrilegious. Thankfully the branching diagrams showing evolutionary relationships, *phylogenetic trees*, are no longer constructed with so much arrogance.

What to do? One alternative is to deny the importance of organism complexity altogether. The apparent increase in complexity of organisms over evolutionary time (if there is such a trend and suitable measure to quantify it) could potentially be due to random drift in organism "complexity space". Therefore there may be no driving force behind complexity increase at all. Here is the basic argument.

To start with, at least moderately simple organisms are going to appear because they are the most likely to self-assemble. When the first self-assembled organisms capable of replication do reproduce, perhaps some of their offspring will be *simpler* than their parents, just by chance. Some of these simpler creatures may even thrive! How could this be? Consider a species about which we know a little, a bird. Perhaps a child manages well without the wings of its parents by foraging on the ground, or maybe its existence in a cave allows it to happily dispense with the complexity of its parents' eyes. On the other hand, perhaps, by chance mutation, offspring of an organism are more complex than their parents. These might survive too. Their parents might not be quite as mobile as their children

who have a tiny new appendage and can now actively flee predators and chase food. Natural variation both up and down the complexity scale is conceivable and examples hark back even to Darwin and Wallace's considerations.

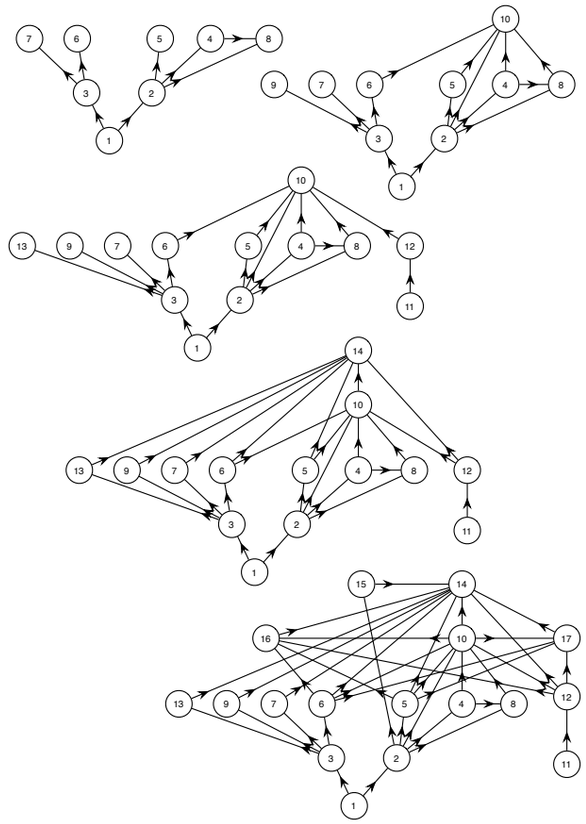
As evolution occurs, some offspring will be simpler than their parents, some will be more complex. Either might survive to continue the evolutionary line. This is possible without any particular driving force towards greater complexity. If that is all there is to it, then complexity increase of organisms over evolutionary time periods is not really very interesting, it just happens due to random variation. Of course that might not be all there is to it. But it is at least a distinct possibility.

Ecosystem complexity

So much for organism complexity. What can be said of *ecosystem* complexity? As evolution continues, do ecosystems increase in complexity? Is there a reason to think they should?

As before, we first need a measure of ecosystem complexity. Perhaps we could count the number of methods for exchanging matter between species, the different types of material being exchanged, or the different chemical reactions occurring. Or perhaps the number of species dependent on one another, or even just the number of species in an ecosystem? These features can all (potentially) be measured. Fortunately, the concept of the *niche* encompasses these phenomena.

Informally, niches are the roles organisms occupy, 'ways of life that lead to the maintenance of species through evolutionary time. Niches take account of the provision of resources (inputs) *to* the organisms; and the provision of resources *by* the occupants of the niche to others outside it (outputs or impacts). If we can identify niches within ecosystems, and their input/output relations, we can describe them in a directed graph. Measuring niche complexity can then be achieved by measuring graph complexity.

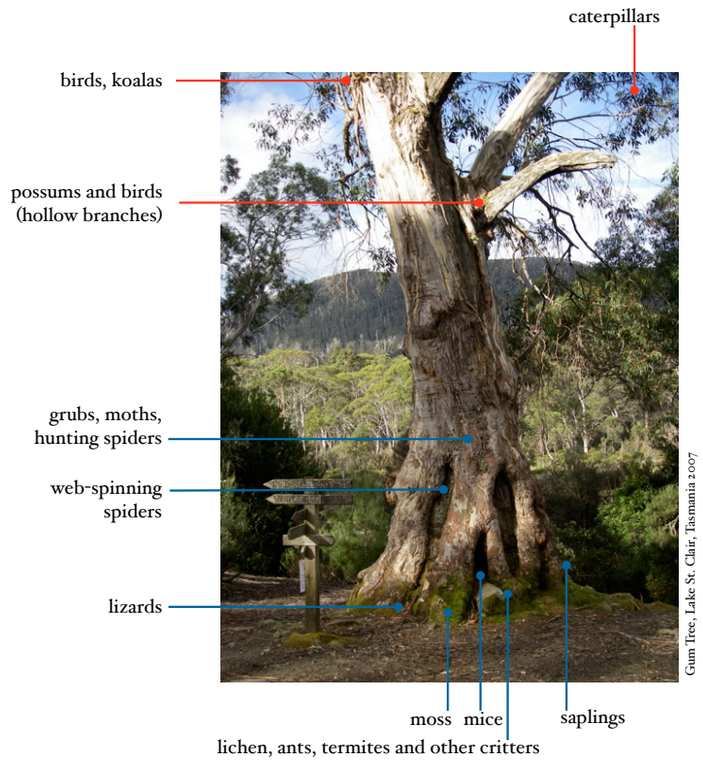


Some sample foodwebs in which nodes represent species and (directed) edges trophic relationships. These graphs illustrate a subset of the interactions between niches and can potentially serve as a proxy in niche-web complexity computations.

Admittedly, it is difficult to gather the data to generate these graphs of niches, but it is not impossible. And in some cases graphs that approximate niche webs have already been produced. The quantification of graph complexity has been attempted in a few cases using information theoretic measures. Without delving into detail, these compute the minimum number of bits required to describe a network of nodes and edges unambiguously. The more bits required, the more complex

the network must be. The question remains, *Does evolution drive an increase in ecosystem complexity?*

Even without detailed measurements, there are some arguments in support of the hypothesis that niche complexity increases exponentially. Each new organism generates the potential for new niches to be occupied by subsequent species. Each new type of body provides new habitat for other species (like bacteria in the human gut or fleas on a dog's back). Each new species changes its environment locally and sometimes globally. For instance, consider global oxygen production by photo-synthesising plants, global CO_2 production by humans, the behaviour of Daisyworld and the Gaia hypothesis. These environmental alterations are performed by *ecosystem engineers*, species that provide habitat by changing the availability of resources or by altering the forces acting on other organisms.



Ecosystem engineers are species that provide habitat by changing the availability of resources or forces acting on other organisms. These are not often modelled in virtual ecosystems — a serious oversight! This Tasmanian eucalyptus tree is a significant ecosystem engineer.

For example, new species' excretions provide new chemicals, their biomass locks up selected materials, changing their availability to other processes. Every new species opens doors to new *kinds* of interaction, new niches for yet more new species. The increase of niche complexity may be an inevitable consequence of evolution generating new species.

An obvious limit applies on the number of species in an ecosystem. This is the carrying capacity of the environment. Sometimes species go extinct too and this might lower the niche complexity of a particular ecosystem. But the extinction of a species leaves at least its original niche, or a new one introduced by its former existence, so the space is open for new species to occupy. Niche complexity leaves the door open for ecological field experiments to test its viability.

Complexity increase in software

How should we assess whether or not a simulation generates increasing complexity? There is no agreement here either. But at least the digital basis of simulations makes it practical to try different measures and see how our code stacks up.

Even trivial simulations can generate new organism types since in the virtual world a single gene changing value from 1.0 to 1.1, as trivial as that is, might conceivably be counted as a speciation event. The concept of a virtual species is difficult to pin down. Can more interesting developments occur?

The hope in software such as Tierra and PolyWorld has been that behavioural complexity will increase. One way it might do so is by hierarchically (or less elegantly, linearly) assembling increasingly complex organism behaviours from the basic possibilities hard-coded at the outset. To some extent, the appearance of the species reported by Ray and Yaeger is evidence of increases in complexity. Yaeger and his collaborators have also seriously assessed the complexity of PolyWorld agent brains over evolutionary time. They analysed these using information theoretic measures and have even driven standard genetic algorithms with fitness functions specifically selecting for information theoretic complexity of agent neural network brains. They evolved complexity in agent brains but this didn't seem to generate equally interesting complexity of agent behaviour.

In PolyWorld and Tierra the niches that appear are primarily related to the interactions between species and a very simple abiotic environment. This simplicity limits the types of niches the simulations might support. Agents lack the ability to construct physicochemical niches. One way to overcome this is to build agents from the same stuff as their environment using artificial chemistry. Then agents may evolve to metabolise abiotic elements in different ways; to eat, excrete and generate new kinds of molecule for use by themselves and other species. For example plants photosynthesise to create sugar, spiders make silk and glue, bees make wax and honey. These new materials fundamentally change the "niche-scape" for many, many species, not just for their makers.

Another aspect of many Artificial Life ecosystems is the single virtual scale at which they operate. For practical reasons it is difficult to simultaneously support a simulation running at levels analogous to physics/chemistry, individual organism interactions and the level of multiple ecosystems. But if explosions in the number of niches appearing with each new species is desirable this would certainly help. The aim would be to support a virtual amoeba metabolising artificial chemical nutrients, while living in the gut of a virtual mouse, who lives in a stable hole in a tree, and is hunted by a snake. Natural food chains often start at the micro-level and proceed upwards in scale, and then dramatically turn downwards as decomposers reclaim even the largest lifeforms for recycling.

Virtual ecosystems are really *interesting*, partly because we still have such a long way to go!

Further reading

Bedau M (2006) *The evolution of complexity*. In: Symposium on the making up of organisms. École normale supérieure.

Bedau MA, Packard NH (1991) "Measurement of evolutionary activity, teleology, and life", In Langton *et al* (eds) Artificial life II. Addison-Wesley, Reading : 431–461

Bronowski, J. (1970). New concepts in the evolution of complexity. Synthese 21, 228–246.

Channon A (2006) Unbounded evolutionary dynamics in a system of agents that actively process and transform their environment. Genet Program Evolvable Mach 7:253–281

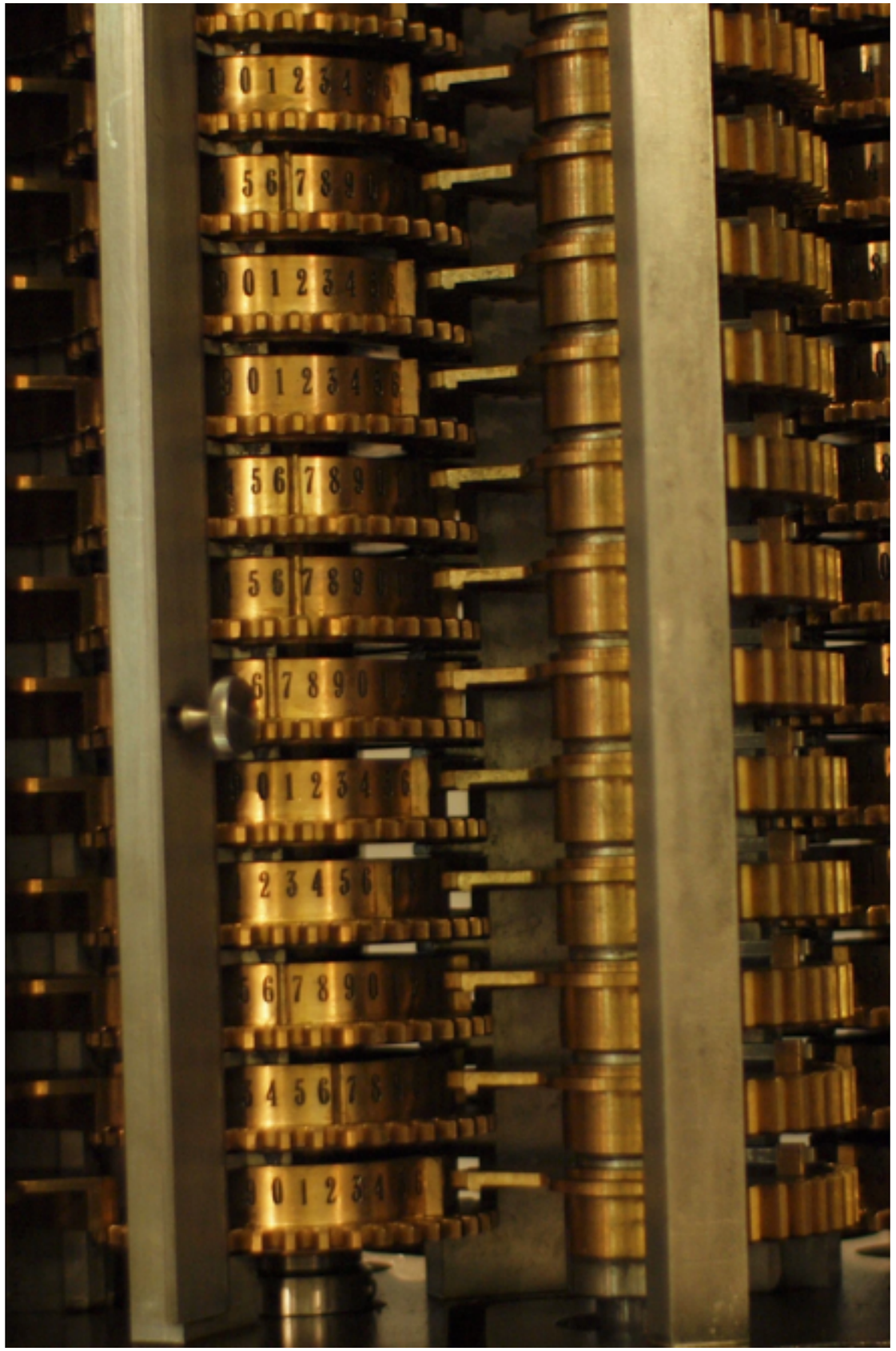
Korb K.B., Dorin, A., (2011) "Evolution Unbound: Releasing the Arrow of Complexity", Biology and Philosophy, Vol. 26: 317-338.

Yaeger, L. S. (2009). *How evolution guides complexity*. HFSP, Vol. 3, No. 5: 328-339.

Chapter 8

Concluding remarks

This is the end of the beginning of *your* work in Artificial Life. If its longevity is any indication, I am not alone in believing this to be the single most interesting field of human endeavour; the technological re-invention of life itself. I hope you have found this introduction helpful and I look forward to reading about your own experiments, discoveries and insights.



Glossary

Abiota
Non-biological components of a region.

Agar
A gelatinous substance derived from seaweed that is often used as a food source on which to grow bacteria in flat, round, glass dishes (petri dishes) in the lab.

Agent
A robot or a computer program component situated in a real or simulated environment that is capable of autonomously perceiving its local area and responding to it.

Artificial Neural Network (ANN)
A computational model inspired by an animal's central nervous system and brain architecture. Typically an ANN consists of several layers of nodes (representing neurons) connected to one another in a network through which signals travel. The network as a whole is capable of machine learning and pattern recognition if designed and trained properly.

Autonomous
A robot or software agent is autonomous if it acts according to its own rules and therefore exhibits a degree of independence from external control.

Balance wheel
A balance wheel is a component of a clock or watch that operates like a rotary pendulum. It is a weighted wheel with a torsional spring, the balance spring, that causes it to rotate in simple harmonic motion.

Biota
The biological entities of a region.

Catalyst
A material whose presence at a reaction site increases the rate at which particular reactions occur, without itself undergoing any permanent chemical change.

Finite State Machine (FSM)
This is an abstract machine that, at any moment in time, is in one of a finite number of alternative states called its "current state". The machine's current state changes to a new state when some prescribed event or condition occurs. A particular state machine is defined by its list of possible states, and the conditions for making a transition from each of these to the next.

Fractal
A geometrical structure with the same statistical properties (e.g. jaggedness) at multiple scales.

Moore neighbourhood
The Moore neighbourhood of a cell on a two-dimensional square grid/lattice, includes cells immediately above, below, left, right and in all diagonal directions. I.e. cells in the direc-

tions of the cardinal points of the compass: North, East, South and West as well as to the North-East, South-East, South-West and North-West are part of this neighbourhood.

Pheromone
A chemical substance deposited into the environment by an animal, especially an insect or a mammal, that causes a change in behaviour of another animal. Ants for instance use pheromones to mark trails or to label hostile invaders.

Protocell
A chemical concoction that might (potentially) act as a precursor to a biological cell. Protocells are often the goal or studied artefact of Artificial Life research in wet chemical laboratories. They typically have some kind of self-assembling membrane demarcating an inside and outside. Some are even motile and interact with one another in surprising ways.

Stack
A software data-structure for storing information that is retrieved in a "first in, last out" fashion. Information is added to the "top" of the stack by pushing it on. Further information continues to be added to the top of the stack by subsequent push operations. To retrieve information from the stack, a pop instruction is executed that returns the most recent value added to the top of the stack and removes it from the data structure.

Stochastic
Based on a probability distribution that may be understood statistically, but which cannot be predicted precisely.

Turing Machine
A model of a hypothetical computing machine that manipulates symbols inscribed on an infinite tape according to a set of predefined rules. The concept was introduced by Alan Turing in 1936. The conceptual machine uses the rules it is given to generate output by processing data that is fed to it as input on the tape. In theory, the Turing machine is capable of executing any algorithm.

Virtual Machine
A software-based implementation of another computing machine that can run within a real computer and executes a program as if it was a physical machine. I.e. the virtual machine is a machine that simulates a real machine.

von Neumann neighbourhood
The von Neumann neighbourhood of a cell on a two-dimensional square grid/lattice, includes cells immediately above, below, left or right of it. I.e. cells in the directions of the cardinal points of the compass: North, East, South and West. Cells that neighbour the current cell along diagonals are not part of this neighbourhood.

Voxels
A voxel is a volume-element, a cubic subdivision of a 3D model in Cartesian space. Voxels are analogous to the pixels of a 2D image.