Evolving Simulated Mutually Perceptive Creatures for Combat

Michael J.T. O'Kelly[1], Kaijen Hsiao[2]

[1] Department of Physics, Massachusetts Institute of Technology, Cambridge, MA 02139
[2] Computer Science & Artificial Intelligence Laboratory, Massachusetts Institute of Technology
Contact: mokelly@mit.edu

Abstract

A fundamental obstacle in evolutionary simulations is the necessity of designing more complex simulations to elicit more complex behaviors. We use a combat-based fitness measure to attempt to circumvent this problem. We have designed a simulation that simultaneously evolves the brains and bodies of creatures for one-on-one combat in a three-dimensional environment with realistic physics. By giving the creatures a rich sensorium, we allow them to react sensibly to each others' actions. We discuss the effective, elegant and diverse simulated fighters that emerge, and examine whether qualitatively greater evolutionary complexity arises.

Introduction

The purpose of a Sims-style (Sims 1994a) simultaneous evolution of creature bodies and brains is to allow the greatest possible flexibility in evolution's solution to a given fitness problem. Sims found that a goal as simple as rapid locomotion elicited a variety of wildly different morphologies, with neural circuitry finely tuned for bounding across the virtual Serengeti.

In (Sims 1994b) the same environment and creature representation was used to evolve participants in a kind of contact sport. Creatures had to grab a box in front of them and then figure our how to keep it away from their opponents. This simple task elicited an evolutionary arms race as each generation's tactics were foiled by the next.

Subsequent work of this kind has gone in two directions. More complex and subtle genetic representations, such as L-systems (Hornby & Pollack 2001) and genetic regulatory networks (Brongard 2002), have been used to evolve articulated robots with realistic physics--but only to accomplish simple tasks such as locomotion. Co-evolution and arms races in warring animats has been considered in, for instance, (Cliff & Miller 1996) and (Floreano & Nolfi 1997), which simulate the chase between predator and prey with highly simplified models of the world.

(Nolfi & Floreano 1998) discuss the circumstances in which arms races may occur, concluding that both environmental and sensory richness encourage this phenomenon. We therefore embed our creatures in a rigorously simulated physical environment and provide them with a full suite of kinesthetic and external senses. Our creatures are then tested on their ability to engage in physical combat. Pairs of creatures are placed in an arena and allowed to fight. A specific part of each body is designated a target; the winner is the first creature to touch its opponent's target.

Sims' competing creatures could see only the box they sought, but were blind to their opponents. Our combat-based fitness measure benefits from the moving fitness landscape of all co-evolving systems and the phenotypic complexity of a Sims-style simulation, while also requiring our creatures to be non-deterministic and responsive in dealing with opponents. We hope to observe an improvement in the resultant complexity of our organisms.

Creature Morphology

As in Karl Sims' work, our creatures are three-dimensional, articulated creatures whose morphologies are determined by directed graphs. Unlike his creatures, ours are composed entirely of spheres, connected by motors that spin along the axis of attachment, much like a wheel and axle. All body parts originate from a root node that also functions as the target for opponents' attacks. Each sphere has a set of sensors associated with it, along with an embedded neural structure that connects sensors to the joint motor via a network of computational neurons.

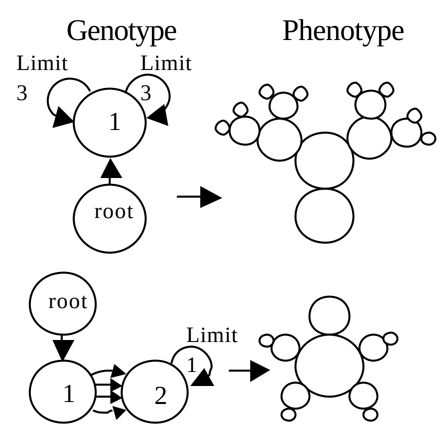


Figure 1: Sample directed-graph genotypes and the resulting phenotypes. Limits show how many more links will be followed.

Examples of two directed graph genotypes and the morphologies they represent are shown in Figure 1. A genotype is translated into a phenotype by starting from the graph's root node and following all of the outgoing links; each time a link is traversed, a new sphere is attached in the phenotype. Links specify the radius of the first sphere in a limb, a scaling factor for subsequent spheres, and other parameters such as the position of attachment. Nodes can be linked recursively back to themselves or in larger cycles, and a single node can link multiple times to its child.

The total number of spheres allowed in any creature is limited globally, forbidding them from becoming too large to simulate. To prevent the representation of infinite-length limbs, nodes have recursive limit parameters that set how many more downstream links may be traversed by the translation algorithm. A separate set of links flagged as extremities can be followed when a recursive limit is reached, so that structures like hands may be attached to the end of recursively-built arms.

These features make our directed graph representation very expressive. Small mutations in the genotype can result in drastic changes in morphology, facilitating evolutionary exploration of the fitness landscape.

Sensors

Each part of a creature's body has sensors that feed it input about its environment so that it can react to changing conditions. One sensor indicates whether the body part is touching either the ground or any part of its opponent. Three sensors indicate the spherical coordinates of the enemy's root node, and three more indicate the spherical coordinates of the nearest enemy sphere. The last two sensors report the position and velocity of the sphere's joint motor. All values returned by sensors are given in the reference frame of the sphere to which the sensor is attached.

In giving our creatures sensors, it would be ideal to provide complete information about the posture and movement of the opponent--information a human would glean from watching a video of a battle. However, there is a tradeoff between having more information available to enable more sophisticated tactics, and keeping the neural structure simple enough for evolution to find useful control structures (Martin 2001). The sensors we used proved more than adequate.

Neural Structure

We want to provide our creatures with the most general possible way of responding to the information provided by their sensors. Ideally, the creature should be able to mull over past events, combine information from throughout its body, and maintain an internal state.

To this end, we use directed graphs to specify our neural circuitry. Our method is comparable to that of (Sims 1994a), with some elaborations. Each node of the morphologic genotype has an embedded graph directing how information will be processed in the phenotype. Nodes in the embedded graphs, referred to as "neurons," represent a specific operation that takes some number of inputs and computes a single output. Links of the graph indicate which outputs are to be fed to which inputs. Self-loops and cycles in the graph represent computational feedback loops, and make our representation Turing complete.

Sensors are represented as neurons that take no inputs. A lone motor neuron in each body sends its input to the attached axle motor. Our motors are velocity controlled, interpreting the motor neuron's signal as an intended velocity; a positive signal indicates counterclockwise motion, while a negative number indicates clockwise motion.

The rest of the neuron types perform computations:

- standard arithmetic operations +, -, *, /
- unary operations sin, cos, atan, log, exp, & sigmoid
- logical operations <, >, & if-greater-than
- integration, differentiation, smoothing & interpolation
- min, max, sum-threshold
- sinewave & sawwave
- constants (fixed output)

The sinewave and sawwave neurons are time-dependent, varying with simulation time. Other neurons, such as the constant neurons, are dependent on evolved parameters in addition to any variable inputs.

At each time step of the simulation, the neurons calculate their outputs in parallel based on their current inputs. It therefore takes more time for the organism to make a complex calculation than one that is simple.

Figure 2 illustrates the neural structure for a simple behavior, a proportional controller. The spherical coordinate denoting the angle in the xy plane of the opponent's target (ϕ) is multiplied by -0.1 and fed to the motor neuron. In simulation, this body part would rotate to keep the opponent's target at a constant bearing.

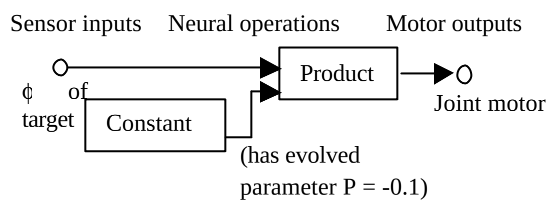


Figure 2: A basic neural structure encoding a proportional controller.

Advanced Neural Structure

The basic neural circuitry provides connections only between sensors and motor neurons on the same body part. In order to allow a creature to coordinate behavior across its body—for instance, to swing a limb in attack or flex away in defense—we want a general framework for the transmission of information from one body part to another.

The root node is the ideal location in our creature for evolving a seat of cognition. It has no motors to control, but it is the most vulnerable part of the body. Its sensors are situated to recognize danger and communicate states to the rest of the body. We therefore allow every neuron in the body to take input from any neuron in the root node.

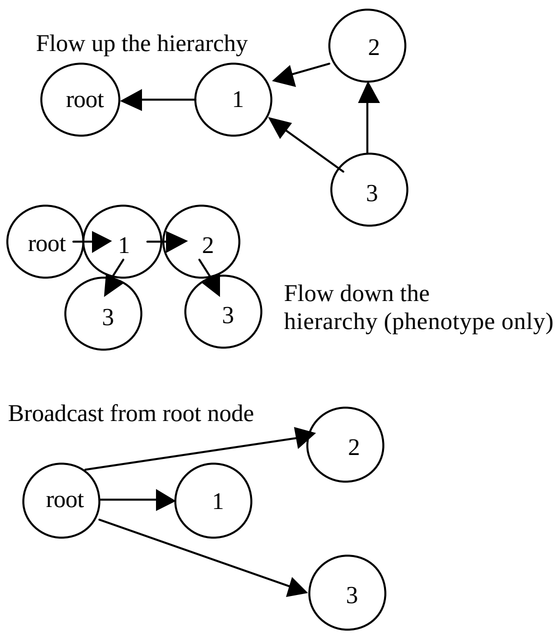


Figure 3: The three modes of communication across the body. Any neuron may draw input from neurons in a child node or in the root node. To take data from a parent node in a way that can be specified unambiguously, special "spinal neurons", which remain unconnected until instantiated in the phenotype, must be used.

Two other modes of linking are provided for the genotype to unambiguously specify how data flows between adjacent body parts. They are illustrated in Figure 3.

Combat

We test our creatures' relative fitness by pitting them against each other in pairs. To create the world in which our creatures are embodied, we use the open-source physics simulation engine Open Dynamics Engine (ODE). ODE models the creatures' bodies, joints, and motors in a physically realistic way, as well as the floor and forces such as gravity and friction. Collision detection and handling is also done by ODE, using a hash space collision detection algorithm for efficiency.

Combat works as follows: two creatures to be compared are embodied in the simulated arena, made to face each other at a fixed distance, and awakened. The first creature to touch its enemy's root node is deemed the winner. Before the contest begins, the creatures are made to relax for a short time without the use of their joint motors, so that they are in a settled configuration when the competition starts. We discovered that without this tweak our creatures evolve only toward extreme height, fighting by falling upon their enemies. Forcing them to relax before combat obligates them to discover a means of locomotion.

We tuned our numerical fitness measure to encourage discovery of basic behaviors such as movement and tracking in early generations. The first creature to hit the other's root node gets the maximum possible fitness value, MAX_FITNESS. The loser is then assigned a fitness value inversely proportional to its distance from the enemy's root node, as a consolation prize for at least having gotten close to its opponent. Without this consolation fitness our creatures found it too risky to approach each other, and never evolved to meaningful engagement. If neither creature hits the other's root node within a fixed amount of time, the game is over and both creatures earn a fitness value inversely proportional to the distance between the two. The actual fitness value is ½ MAX_FITNESS * (initial distance – final distance). Fitness can never be less than zero, however, so if a creature wanders off into the distance, its fitness will be zero. This is to prevent an otherwise fit creature from being eliminated because its opponent went romping off to the middle of nowhere.

Evolution

Selecting Parents

In order to apply evolutionary pressure to our creatures, we require that those that are better at combat reproduce with greater likelihood than those that are worse. However, in our combat-based fitness measure, and in fact any fitness measure derived from direct competition, it is possible for Player A to beat B, B to beat C, and C to beat A. How do we determine who is most deserving?

Since our fitness measure assigns a numerical fitness value to both participants in a fight, an obvious method would be to pair off every creature in a round-robin tournament, and let those with the greatest accumulated fitness reproduce. However, simulated combat runs very slowly, requiring ~1e10 floating point operations per test. A round-robin tournament requires N^2 tests, which becomes untenable with reasonable population sizes.

Instead, we perform selection via miniature round-robin tournaments, a variation on traditional tournament selection. A handful of creatures (we used a tournament size of 4) are picked at random. This subset of the full population plays a round-robin tournament with all possible pairings, and the winner is the player with the greatest accumulated fitness. That player is handed over to the algorithm creating the next generation of players.

This selection algorithm is a good compromise between a desire to rank players fairly and the constraint of reasonable runtime. In essence, any player that can defeat at least three of its peers is considered good enough for reproduction.

(Sims 1994b) discusses a variety of other approaches such as competition between species or against the previous generation's champion. Brief experiments showed little qualitative difference in our results.

Mutation

Directed graphs lend themselves to elegant mutation operators. A genotype selected for mutation may have nodes and links randomly added and removed, or the targets of links randomly changed. Parameters such as scaling factor and recursive limit may be randomly changed. The effect of mutation on the phenotype can be as small as a slightly larger radius and as large as a completely altered morphology. The embedded neural graphs are subject to a similar mutation operation.

Crossover

Our general directed graphs do not have as natural a means of crossover as would, say, a tree graph. The presence of cycles makes it difficult to choose a particular subset of a graph for transplantation. Instead, an artifact of our implementation is used to align parts of two parent graphs by similarity. Each node and link in a genotype is given a unique ID upon its creation. The descendants of a particular genotype also inherit the ID numbers of their components. This results in the genotypes of related creatures having the same ID's for components of similar function.

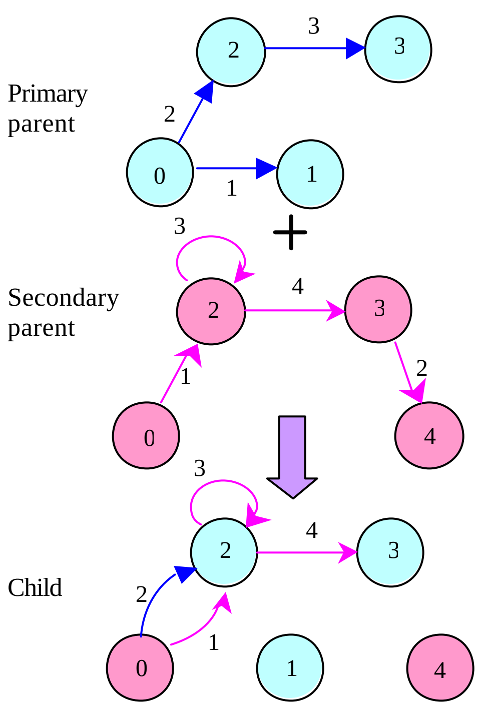


Figure 4: Crossover. Parent genotypes are lined up according to link and node ID's, and some material from the secondary parent is randomly chosen to replace that in the primary.

The crossover operation is illustrated in Figure 4. To perform this operation, a primary parent and secondary parent are selected from the population. Up to half of the nodes and links of the secondary parent are selected at random to be transferred. If a selected node has the same ID number as a node already present in the primary parent, that node is overwritten, along with its underlying neural graph. If the ID is not already present, the node is simply added to the genotype. Links with already present ID's are likewise overwritten, possibly resulting in changes to the graph's topology. Links with distinct ID's are added in the same manner as with nodes.

If the sexually reproducing parents share a common ancestor, as is likely in our small populations, this form of crossover is less brittle than a completely random exchange of genetic material would be. This allows innovations discovered by two different creatures to be shared; in particular, improvements to the neural circuitry are easily shared.

Genotype Validation and Garbage Collection

Before a genotype resulting from crossover or mutation is introduced to the general population, it is checked for physical validity. A genotype could specify a phenotype that self-overlaps, which is forbidden by our physics model. Invalid offspring are destroyed and the reproduction operators are applied again until a valid offspring is discovered.

After mutation or crossover is performed, genotypes are often left with nodes that can no longer be reached from the root node, and therefore are not expressed, as well as links that no longer point to valid nodes. Changes to the directed graph topology will also make many neural connections invalid, such as when one node's neuron tries to draw input from another node that is no longer its child.

The garbage collector works in two stages. First it identifies all unreachable nodes and hanging links, and removes them from the genotype. Then it checks all the remaining neurons to verify that their inputs remain valid. Invalid inputs are reassigned, drawing randomly from the set of all currently valid inputs.

Putting It All Together

We can now describe a complete evolutionary experiment. An initial population is created by randomly generating small directed graphs, checked for physical validity. We use a typical population size of 40. To form subsequent generations, we select 40 parents using the tournament selection rule described above. Duplicate selections are allowed; indeed, the best fighters are usually selected many times. Half of the parents reproduce by mutation, and the other half are paired off to reproduce by crossover, trading off the roles of primary and secondary parent.

The process of selection and reproduction is iterated as many times as desired; our longest runs have about 400 generations.

Results

Our evolutionary runs resulted in a wide variety of successful combat strategies and methods of locomotion.

Most of the creatures we discuss emerged between 20 and 100 generations. While many runs resulted in fairly homogeneous populations by the time they were stopped, each run produced completely different final creatures. A selection of the most successful creatures is shown in Figure 5.

Our creatures typically discovered a successful method of locomotion early on, since spherical structures and the absence of motor joint limits made it easy to roll. The first motion discovered in most evolutionary runs was spinning rapidly in place—an easy way to protect the root node, but not useful for attack.

Usually by the 10^{th} generation a consistent method of locomotion was discovered that remained a theme throughout the rest of the experiment. Panel 1 shows a pair of worms that arch their backs and roll toward their opponents. Panel 2 shows a pair of "breakdancers" that hopped back and forth rapidly from one end to the other; in the frame shown, one is completely in the air while the other has one end on the ground. The breakdancers were rare in that they learned to track their opponents despite being hopping creatures. Behaviors could usually be grouped into wild, defensive behaviors that reacted little to opponents, and more deliberate behaviors tuned to move towards the opponent wherever it went.

Panel 4 shows a creature making a successful adaptation to trump its peers. In a population of creatures that use large defensive arms to shield and attack, the "Spanker" evolved a long extremity that sits inactive until its opponent touches one of its body parts. The creature then swings its long arm around to smack its opponent's protected target. This improvement led the "spanker" to immediate domination in the next generation.

Panels 3 and 5 show more advanced (~60 generations) creatures combining solid attack and defense. The "Proboscis" creatures use a long thin probe both for a cautious rolling attack and as a pivot to sweep at its opponent with its whole body. The "Buggy" rolls toward its opponent, has a front bumper-shield to protect its root node from being casually hit, and uses a mace-like arm to hit an opponent's protected root node.

Our most successful creature is shown in Panel 6. We named it "The Hedgehog" because of its prickly structure and whole-body rolling attack. The Hedgehog has four interchangeable limbs. Two of them touch the ground at any given time, rolling in the direction of the opponent, while the remaining two twirl in the air. The Hedgehog is good at tracking an opponent and its limbs provide some passive defense. What's most impressive, though, is that the limbs constantly move to keep the vulnerable root node as far from the enemy as possible. In response to a sudden attack the Hedgehog's entire body flexes to flip its root node away from harm and land its upper limbs on the attacker's body.

The Hedgehog is also remarkable for the simplicity of its representation. Its genotype is specified with just two nodes, linked recursively; and its neural circuitry consists of just one link between an enemy positional sensor and a motor neuron. (No other neurons are used to any effect.) The Hedgehog's sophisticated behavior is entirely emergent from the geometry of its body and the local interaction of sensors and motors.

Some of our more ambitious intentions did not pan out. With experiments requiring days to simulate a population of just 40 individuals to 400 generations, it remains unclear whether our combat-based fitness measure allows ever-escalating evolutionary complexity or if the plateau has just not been reached. Population diversity tended to bottom out after an initial period of morphological variation lasting around 100 generations.

(Nolfi & Marocco 2002) suggest one possible explanation. They argue that neural structure beyond simple sensory-motor coordination is difficult for evolution to discover because two necessary components--useful internal variables, and connections from them to sensory-motor flow--must arise simultaneously and spontaneously, since no advantage to the creature comes from either alone. Our neural mutation operators were not designed with facilitating this sort of discovery in mind. Evolution of complex neural structure might have consequently required many more generations than a comparably complex morphology. This agrees with our observation that the use of advanced neural structure was surprisingly rare, given the ease of its discovery by our mutation operators.

Also, the fact that fitness depended explicitly on the other members of the population had the unanticipated effect of often encouraging our creatures to be opportunistic. Rather than evolving to outsmart the best of their peers, they instead developed strategies to take advantage of weaklings. A significant fraction of each generation was crippled by mutation or crossover gone awry, and the most successful creatures were those that dealt rapidly and savagely with the defenseless.

These problems are not insurmountable. The first might be handled by designing a new neural representation or reproduction operator, or by evolving one as in (Teller 1996). (Nolfi & Floreano 1998) discuss methods of tournament selection that may better promote arms races. What is most needed, however, is longer experiments on larger populations. (Cliff & Miller 1995) distinguish open-ended from cyclic evolution by testing whether the latest generation of individuals can reliably defeat prior generations. Though cyclic behavior was not observed in our short-lived experiments, neither can open-ended evolution yet be confirmed.

Videos and source code for our experiments can be found online at www.simons-rock.edu/~towk/alife/bubblegene.html .

Conclusion

We successfully evolved creatures for combat exhibiting diverse morphologies and behaviors. Our creatures engaged in an evolutionary arms race, discovering feints, attacks, and dodges once the simpler skills of spinning in place or moving straight forward no longer sufficed. We conclude that our reasons for choosing a combat-based fitness measure—it allows the creatures themselves to determine their fitness landscape, and it forces them to react sensibly to opponents' actions—were validated.

Another facet of our evolutionary framework is the complex behaviors and strategies that can emerge from very simple structures. The best example of such emergent behavior was seen in the Hedgehog, a creature that was able to track its opponent, actively defend its target node by rotating it out of reach of its opponent, and continually attack its opponent with flailing arms. An observer would expect that the Hedgehog must possess a complex internal structure to demonstrate such a sophisticated strategy, whereas in reality all of the above behaviors emerged from a staggeringly simple design.

Acknowledgements

This research was supported by the Fannie & John Hertz Foundation and by the National Science Foundation.

References

Bongard, J. C. 2002. Evolving Modular Genetic Regulatory Networks. In *Proceedings of the IEEE 2002 Congress on Evolutionary Computation (CEC2002)*, vol. 2, pp. 1872-1877.

Cliff, D., & Miller, G. F. 1995. Tracking the Red Queen: Measurements of Adaptive Progress in Co-evolutionary Simulations. In *Advances in Artificial Life: Proceedings of the Third European Conference on Artificial Life*, Berlin.

Cliff, D., & Miller, G. F. 1996. Co-evolution of Pursuit and Evasion II: Simulation Methods and Results. In *Animals to Animats IV: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior.*

Floreano, D. & Nolfi, S. 1997. God Save the Red Queen! Competition in Co-Evolutionary Robotics. In *2nd Conference on Genetic Programming*, San Mateo, CA.

Hornby, G. S. & Pollack, J. B. 2001. Evolving L-Systems to Generate Virtual Creatures. *Computers and Graphics*. 25:6, p. 1041-1048.

Martin, M. C. 2001. The Simulated Evolution of Robot Perception. Ph.D. thesis, School of Computer Science, Carnegie Mellon University.

Nolfi S., & Floreano D. 1998. Co-evolving Predator and Prey Robots: Do 'Arm Races' Arise in Artificial Evolution? In *Proceedings of Artificial Life IV*, 311-335.

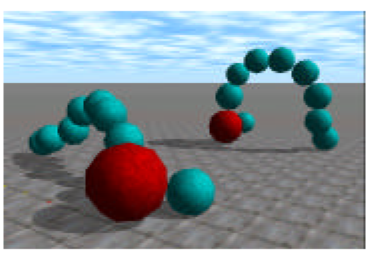
Nolfi, S., & Marocco, D. 2002. Evolving Robots Able To Integrate Sensory-Motor Information Over Time, In *Biologically Inspired Robot Behavior Engineering*, Berlino, Springer-Verlag.

Sims, K. 1994a. Evolving Virtual Creatures. *Computer Graphics*. *SIGGRAPH Proceedings*, 24-29.

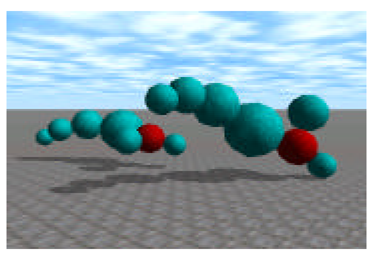
Sims, K. 1994b. Evolving 3D Morphology and Behavior by Competition. In *Proceedings of Artificial Life IV*, p. 28-39.

Teller, Astro. 1996. Evolving Programmers: The Co-evolution of Intelligent Recombination Operators. In *Advances in Genetic Programming 2*. Cambridge, MA: The MIT Press.

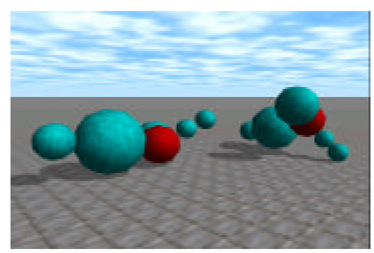
Figure 5: A selection of evolved creatures



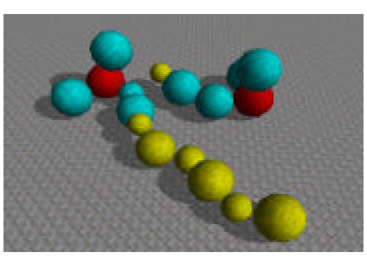
Panel 1: Worms



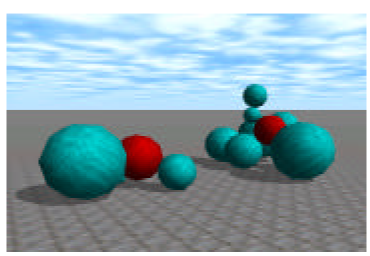
Panel 2: Breakdancers



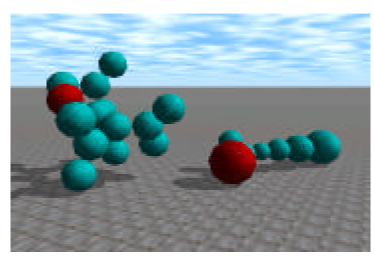
Panel 3: Proboscis



Panel 4: Spanker (left)



Panel 5: Buggy with Mace (right)



Panel 6. Hedgehog (left)