CSCI444/944

Week 13

Evolutionary Robotics

Generativity

Real engineers use same components (modules) in many different solutions: engines, wheels, ...

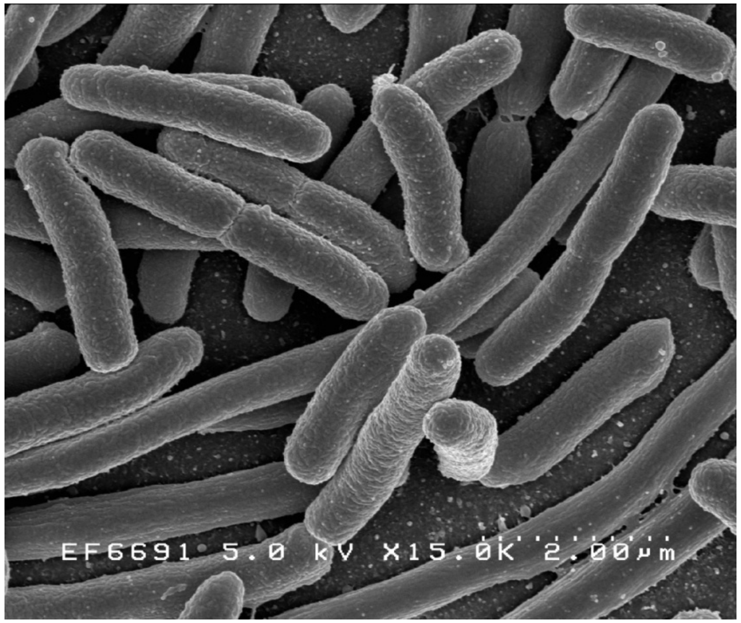
In a more subtle way, nature does too: hearts, limbs, ...

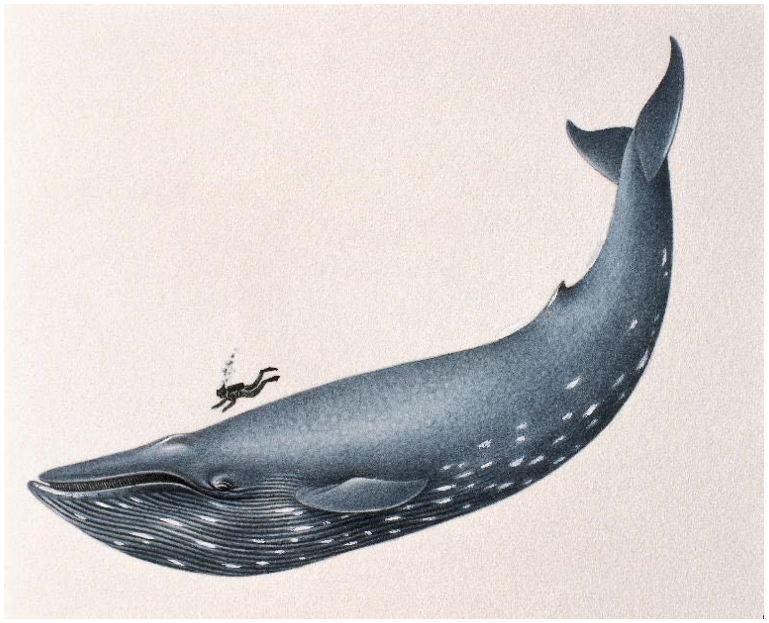
Generativity

- Nature doesn’t directly mutate or recombine modules; instead, the code (DNA) gets recombined/mutated and then generates the biological entity.
- From an engineering perspective, this sort of approach supports much larger, more complex structures.
- Elementary building blocks (cells) are organized to build bigger more complex entities.
- Evolution changes the function and the organization of these building blocks.

Generativity

The same process of generativity allows the development of single cell organisms and multi-cell organisms.





Evolutionary Robotics

- Attempts to elucidate principles of evolution to robots
  - Build models of self-replicating organisms
    - Computational cost limits physical fidelity of the model.
    - Digital or chemical models
  - Mutation creates variation in populations
  - Reproduction can be sexual or asexual
  - Ability to (out) reproduce its genome is the usual fitness measure
    - For some research, other fitness measures are used.

Evolutionary Robotics

Primary goal:

- To develop <u>automatic</u> methods for creating intelligent autonomous robot controllers.

Definition (Wikipedia):

- Evolutionary robotics (ER) is a methodology that uses evolutionary computation to develop controllers for autonomous robots.

Evolutionary Robotics

- Not to be Confused With Evolutionary (genetic) Computing
  - A Search Technique inspired by biology
  - Points in search space represented as “genomes”
  - Crossover produces new points in search space
  - Mutation ensures variety
    - Ensures more of search space is sampled
  - Fitness function determines which subset of population become progenitors
  - Larger populations increase coverage of space.
  - Search usually walks through “invalid” points

Evolutionary Robotics

- Central goal: start with a critical mass then achieve *higher complexity at lower cost*
- Only possible when design of controllers & construction of structure is fully automatic
- In real evolution, brain & body co-evolve in “a long series of mutual adaptations”

Need for ER

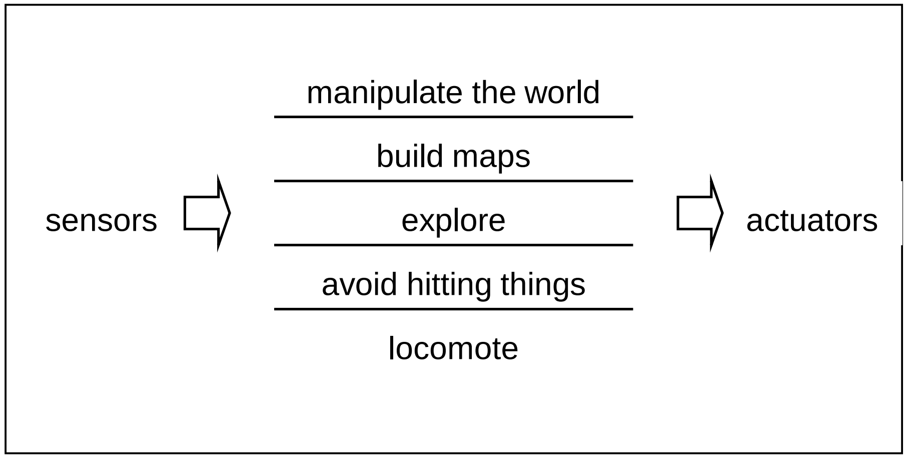
- Robotic systems and the environments into which they are placed increase in complexity, and with it:
    - the difficulty of programming their control systems to respond appropriately
- Difficulty can increase to the point where it becomes impracticable to foresee every possible state of the robot in its environment.
- Purpose of ER is to find adaptable robot controllers for behaviours which are hard to implement by hand

Evolutionary Robotics

ER consist of:

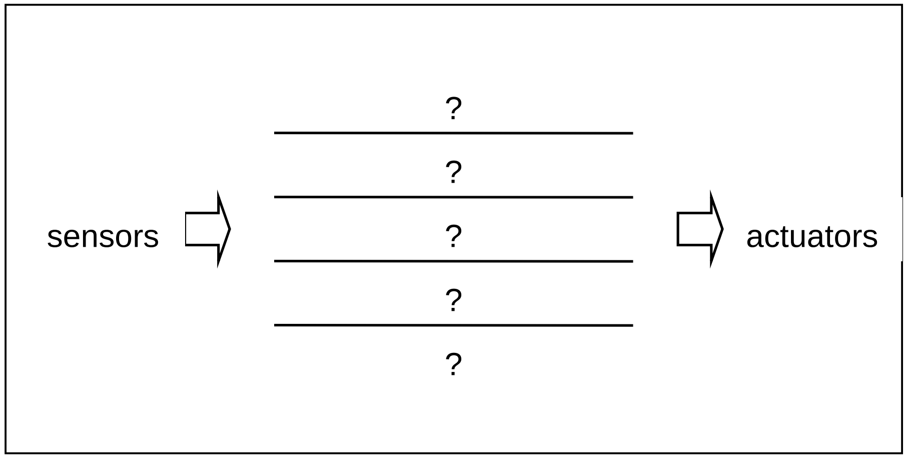
- **Evolutionary Controllers:**
  - The “brains” of a robot.
  - Controls the action of actuators based on the sensor inputs.
- **Evolving Morphological Structure**
  - Body of a robot

Evolutionary Controllers



behavior-based robotics

[Brooks, 1986]

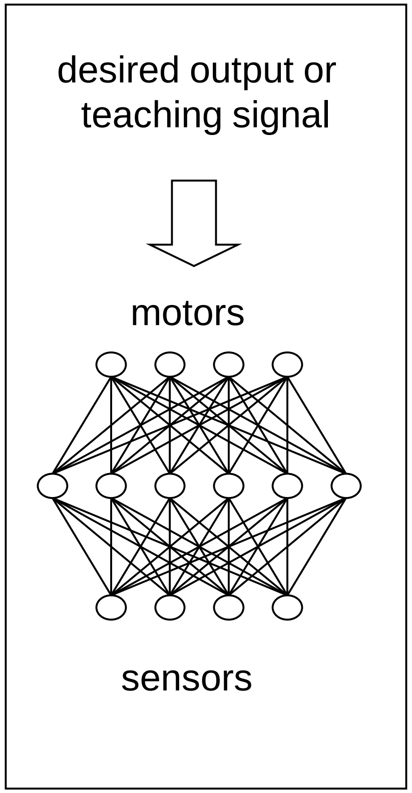


evolutionary robotics

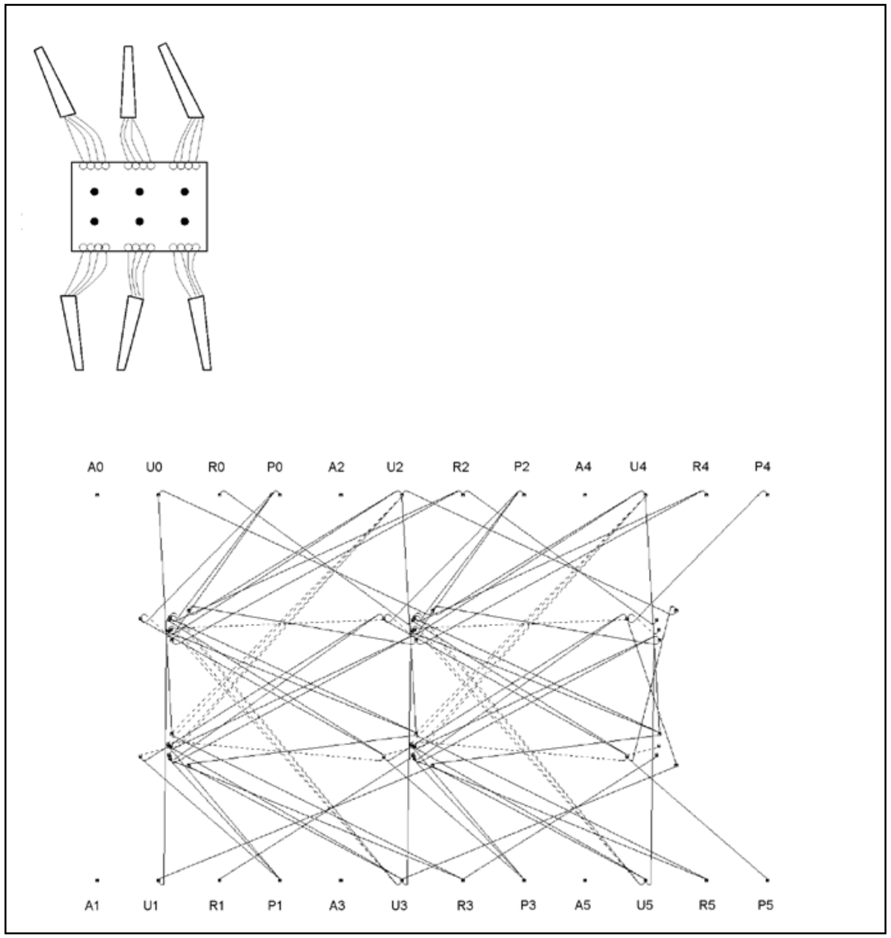
Stefano Nolfi & Dario Floreano, 2000

Evolutionary Robotics

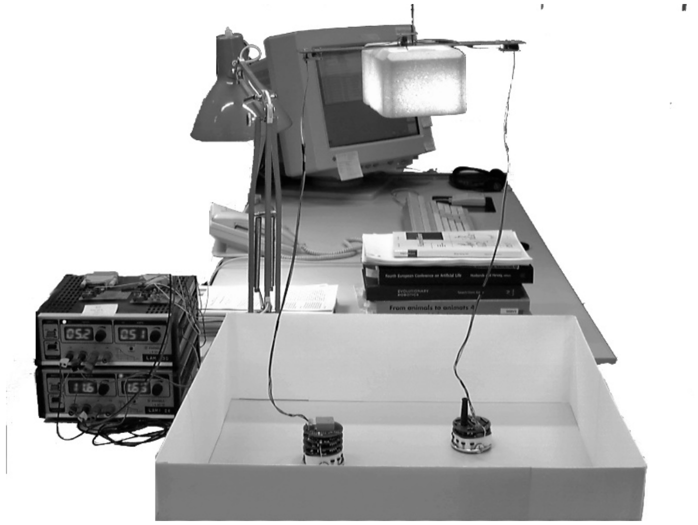
We have already seen some methods that allow robots to adapt and change behaviour i.e. by learning (learning to walk):



Supervised vs. Unsupervised learning

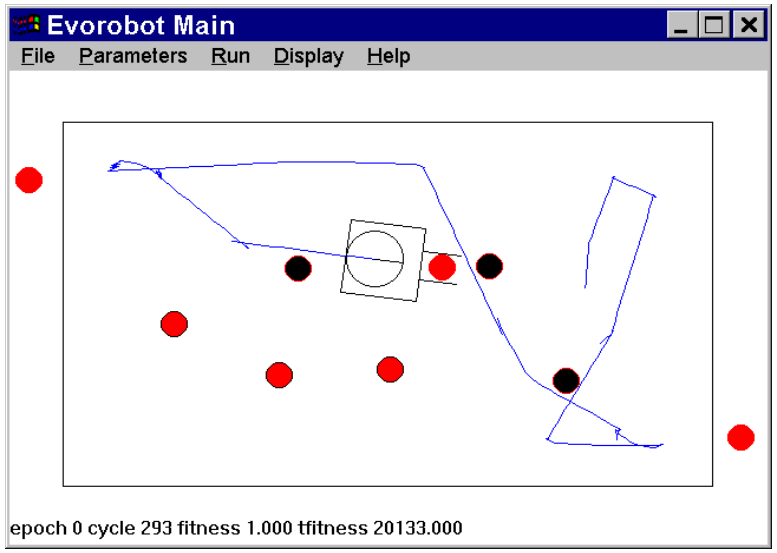


Ho to Evolve Robots



Direct evolution on the real world

[Floreano and Nolfi, 1998]



epoch 0 cycle 293 fitness 1.000 tfitness 20133.000

Evolution in simulations, then test on a real robot

[Nolfi, Floreano, Miglino, Mondada 1994]

Survival of the Fittest

- Each robot controller competes in an environment to perform the task for which the robots are being evolved.
- Evolution creates populations of randomly configured robots.
    - These are not random robots. But the configuration of a robot involves a degree of “chance”. This chance is undirected (not guided by a fitness function) and hence this is random.
    - Survival of the fittest is guided by a fitness function thus providing a direction for improvements of randomly generated candidates.
- The controllers in the better performing robots are selected, altered and propagated in a repeating process that mimics natural evolution.

Evolution

1. **Initiation** – Initial population may be random, blank, or seeded with some prior knowledge in the form of solutions we think are good starting points.
2. **Selection** - measure the performance of each solution in the population. The performance, termed "fitness", captures the merit of the design with respect to some target performance.
3. **Variation** - Select better solutions (parents) and use them to create a new generation of solutions (offspring). The offspring are random variations of the parents, created through operators like mutation, crossover and recombination.

Step 2 and Step 3 is repeated generation after generation until good solutions are found.

Early work on evolutionary Robotics

- Early work on ER revealed that the morphology and the controller should evolve at the same time similar to cellular life.
  - Co-evolution of controllers and morphology

  https://www.youtube.com/watch?v=Wl5rRGVD0QI
- The concept of evolution has been studied on simulations of digital organisms. eg Tierra and Avida and other Tierra inspired work.

  https://www.youtube.com/watch?v=apANCQmPmL0

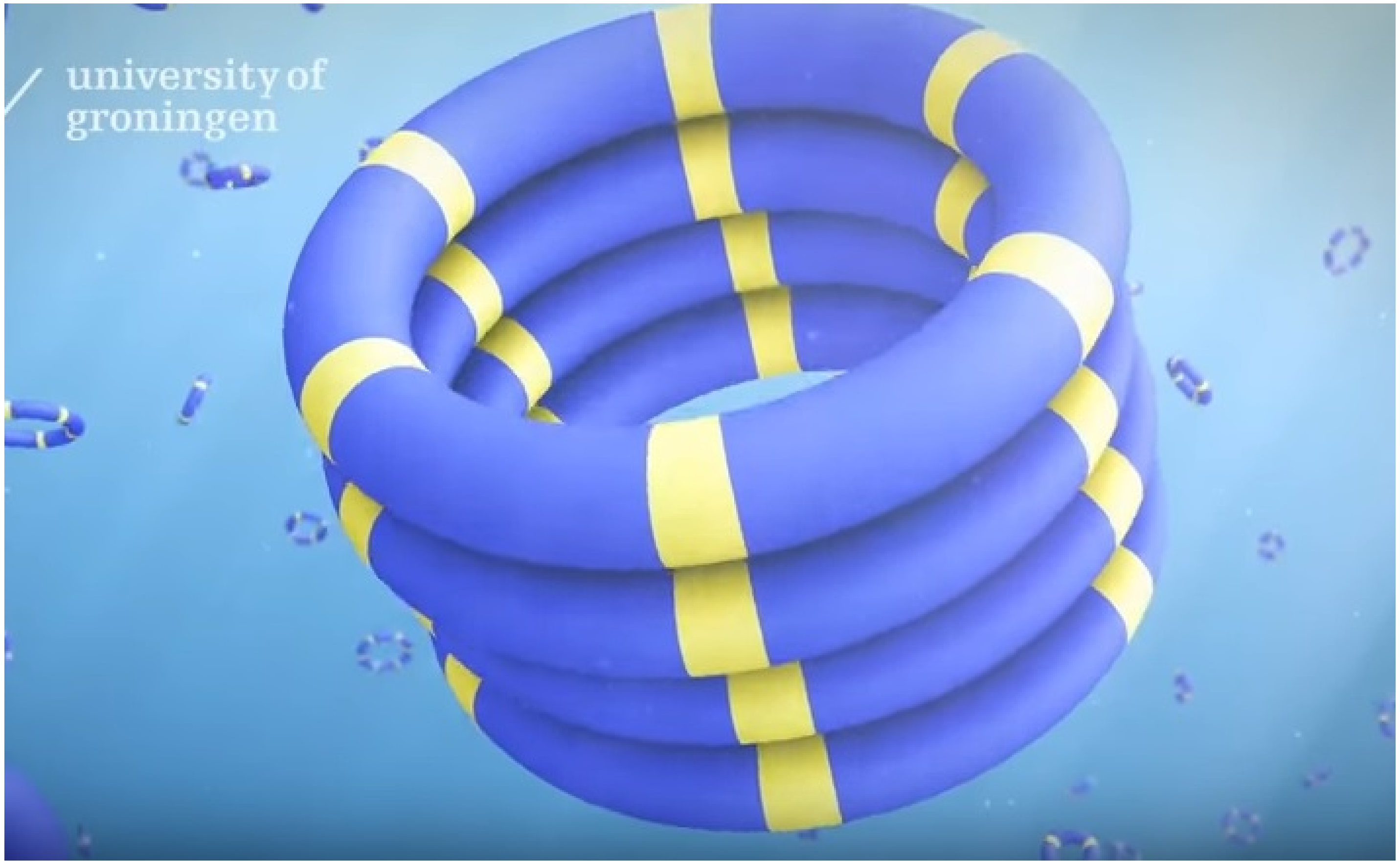
Example: Spontaneous Self Replication

Start with simple building blocks that can react with other building blocks to form compound building blocks.

This initially results in the formation of many different cyclic molecules.

These cyclic molecules continuously exchange components between each other, to give a mixture that is at equilibrium.

Spontaneous Self Replication



Self replicating molecules

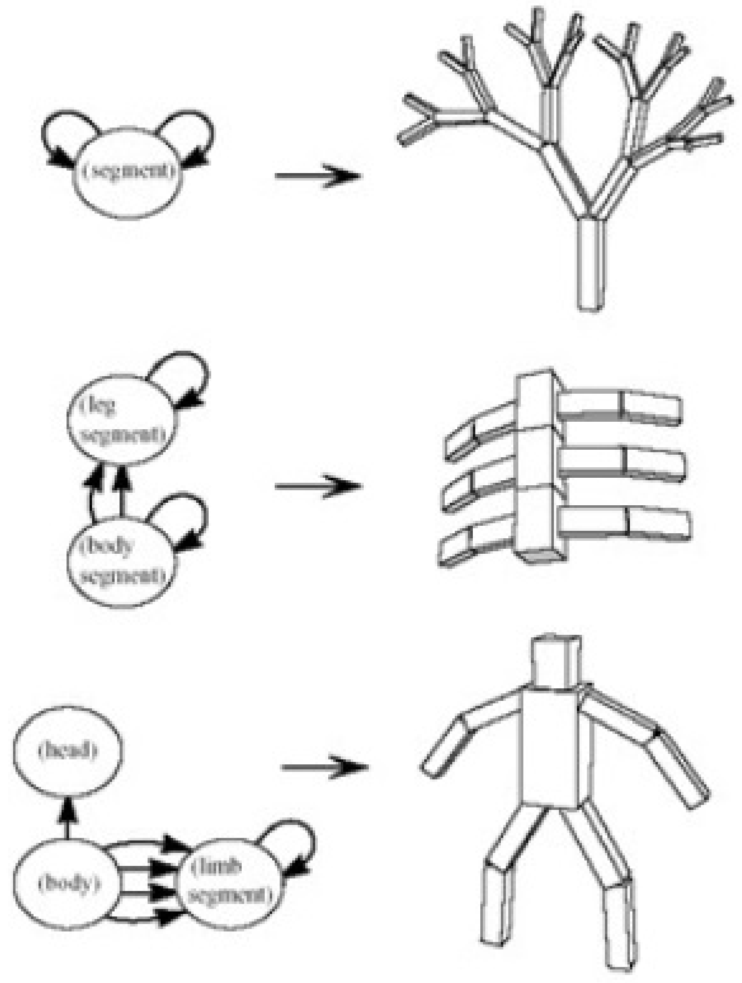
https://www.youtube.com/watch?v=w2lqZL153JE

Evolving Morphology

- Morphology (body) of robots can be evolved like controllers.
- Body of robots made of small units which can be detached / attached / reconfigured.
- Alternatively, robots can be produced using rapid prototyping equipments.

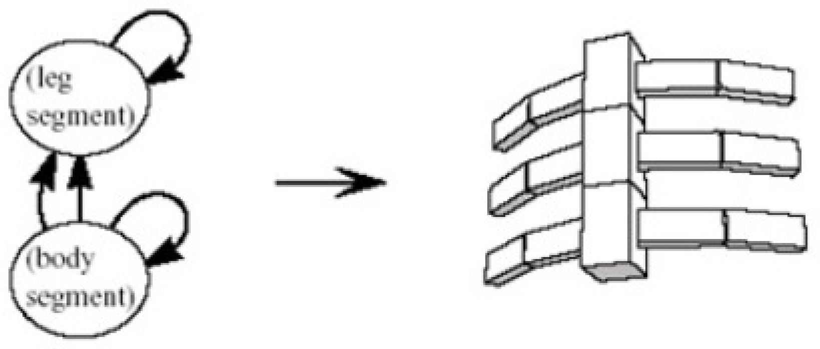
Karl Sims Evolved Virtual Creatures (1994)

- Virtual creatures
- Issues:
    - Creature representatior
    - Creature control
    - Physical simulation
    - Behaviour
    - Evolution
    - Results



Karl Sims Evolved Virtual Creatures (1994)

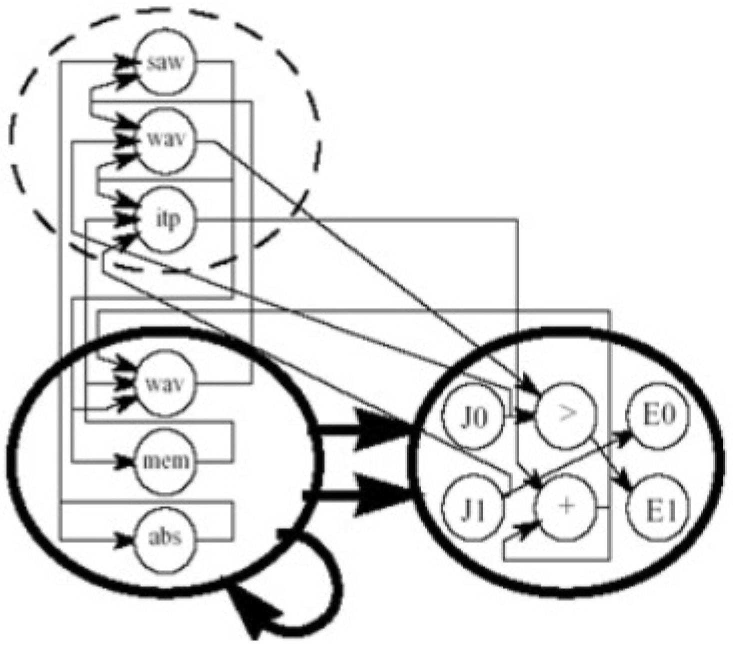
- Based on genetic Algorithms (GAs)
- Darwinian survival of the fittest
- Genotype vs Phenotype



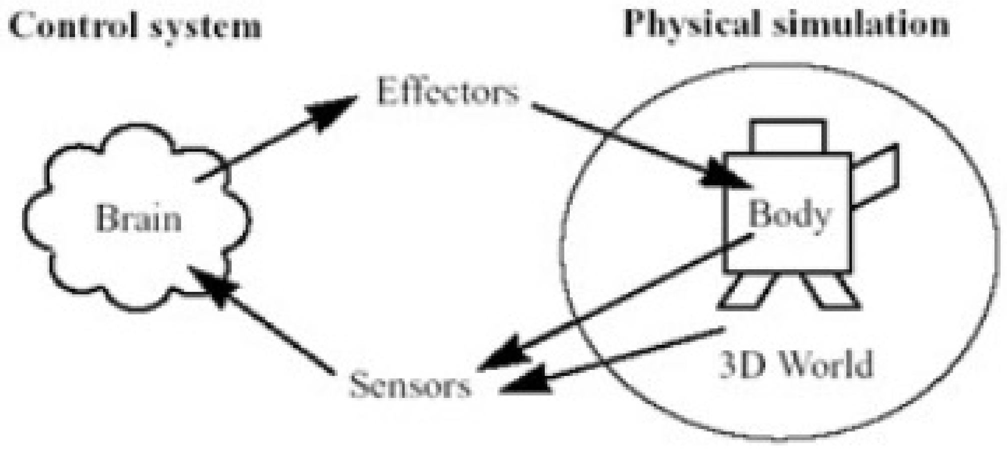
Genotype Phenotype

Karl Sims Evolved Virtual Creatures (1994)

- Directed graph where nodes contain:
  - •Dimensions
  - •Joint type
  - •Joint limits
  - •Recursive limit
  - •Neurons
  - •Connections
  - •Child node:
    - •Position
    - •Orientation

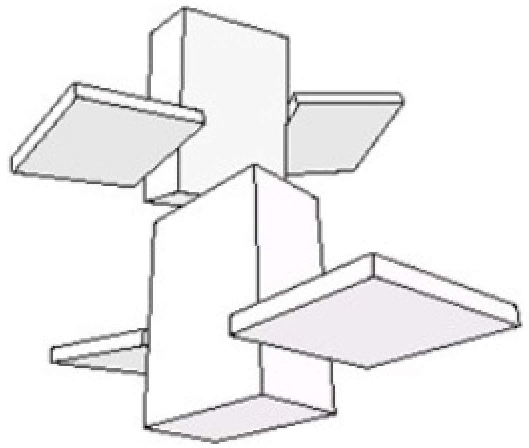


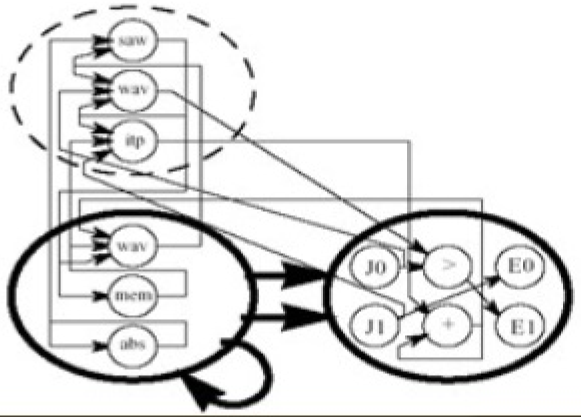
Karl Sims Evolved Virtual Creatures (1994)

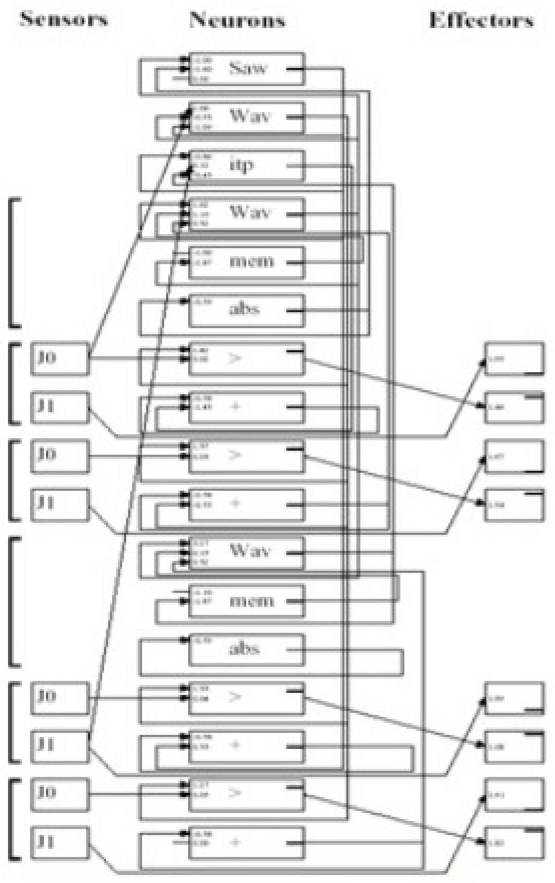


- Brain:
  - Directed graph of neurons
- Effectors:
  - Applied at joints as forces or torques between muscle pairs

Karl Sims Evolved Virtual Creatures (1994)







- Combining Control and representation

Karl Sims Evolved Virtual Creatures (1994)

- Sensors:
  - Joint angle sensors
  - Contact sensors
  - Photo sensors
- Neurons:
  - Provide different functions
  - Sum, product, abs, max, sin, cos, etc...
  - Output vs Input
    - Input dependent functions
    - Output dependent functions

Karl Sims Evolved Virtual Creatures (1994)

Physical simulation:

- Collision detection
    - Bounding box + pair specific
  - Collision response
    - Impulse + penalty springs
- Friction + Viscosity
  - for simulating water and land

Karl Sims Evolved Virtual Creatures (1994)

Behaviours:

- Swimming
- Walking / limping
- Jumping
- Following (land or water)

- Fitness function:
  - Evaluated at each step
  - Preferred methods become weighted

https://www.youtube.com/watch?v=JBgG_VSP7f8

Karl Sims Evolved Virtual Creatures (1994)

Evolution:

- Create initial genotypes
  - From scratch
  - From pre-evolved creatures
- Calculate survival ratio
- Evaluate fitness and kill weaklings
- Reproduce the fittest
- Repeat the above

Karl Sims Evolved Virtual Creatures (1994)

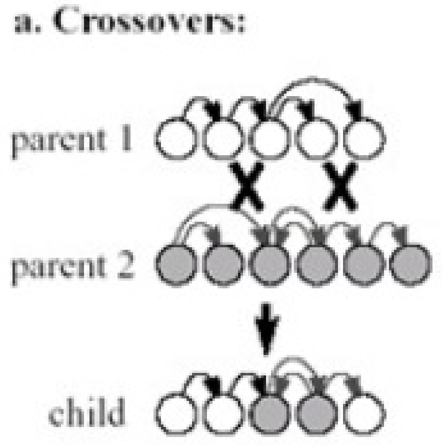
Evolution:

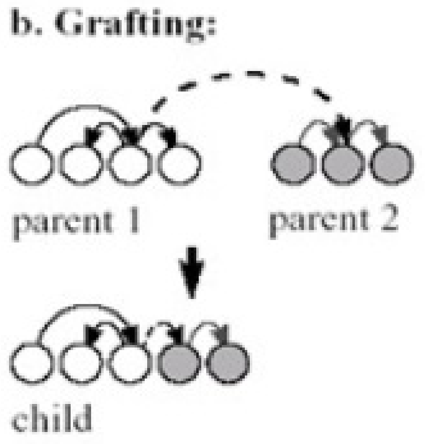
- Directed graph mutation
  - Node parameters
  - Adding random nodes
  - Changing connections
- Mutation frequency scales inversely with graph size.

Karl Sims Evolved Virtual Creatures (1994)

Mating directed graphs:

- 40% asexual
- 30% crossover
- 30% grafting.





Karl Sims Evolved Virtual Creatures (1994)

Performance (CM5 – 32 processors)

- Population of 300
- 100 generations
- 3 hours

Karl Sims Evolved Virtual Creatures (1994)

Results:

- Swimmers
- Paddlers tail-waggers
- Walkers
- Lizard like critters
- Pushers/pullers
- Hoppers
- Followers
- etc

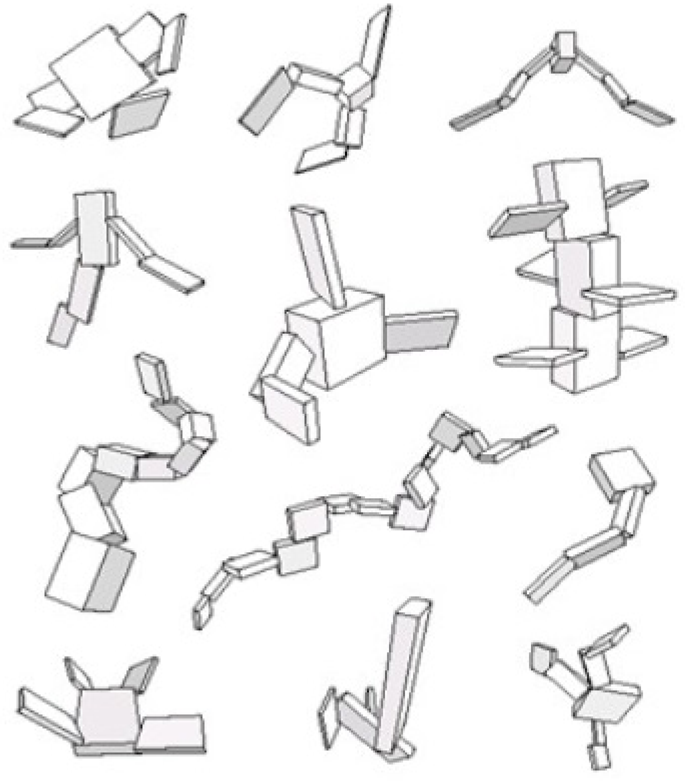
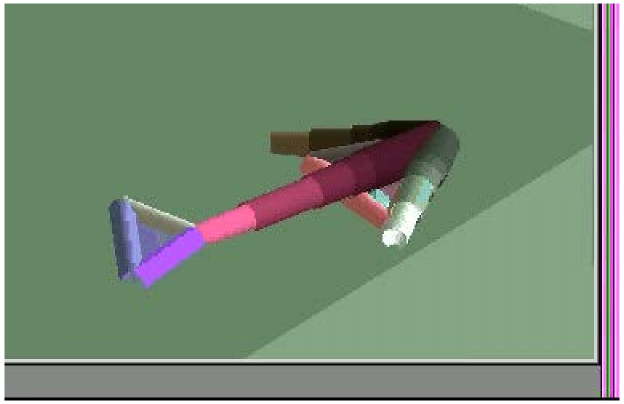
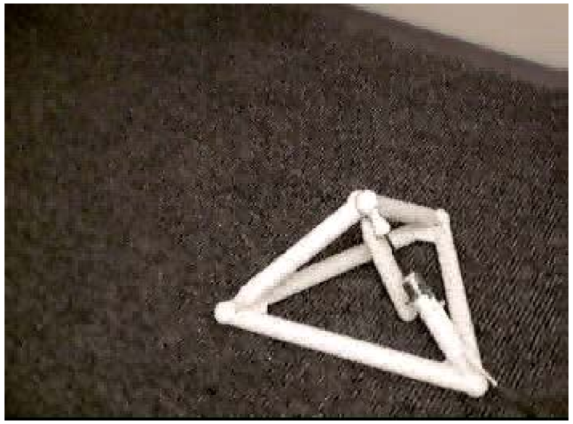
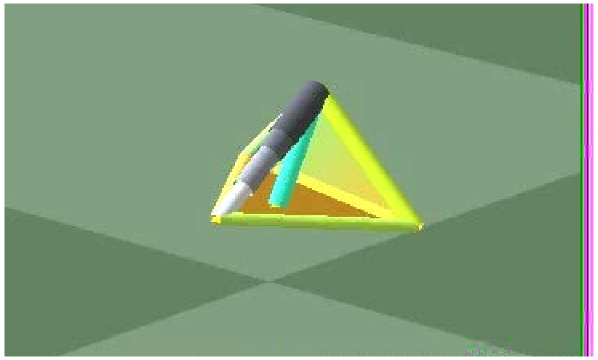
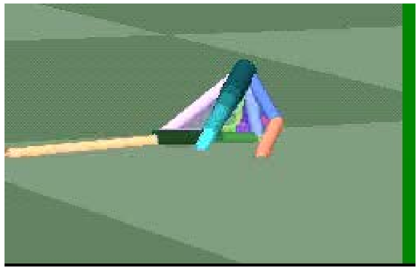


Figure 6: Creatures evolved for swimming.

Karl Sims Evolved Virtual Creatures (1994)

https://archive.org/details/sims_evolved_virtual_creatures_1994

Simulations

- Evolving control systems in simulation provide a solution to prevent hardware damage and provides much faster evaluation than real time speed.
- But, simulation has its own drawbacks:
  - Absence of noise from external environment, imprecise readings and motor control
  - Experiments conducted with high levels of noise – Solutions may evolve that rely on the excessive noise.
  - Simulations may overestimate efficiency (i.e. speed) of a physically implemented robot since not all physical characteristics can be simulated without compromising the speed benefit of simulations.

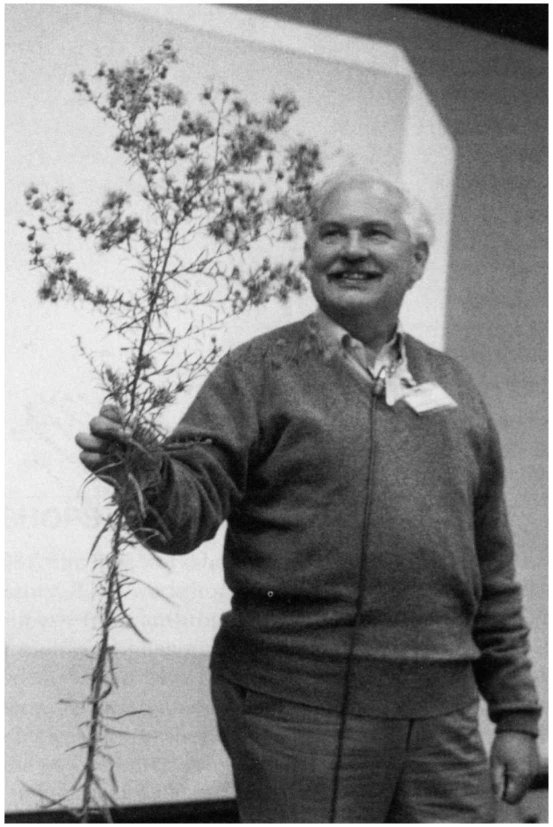
Body/Brain Co-evolution: Simulator Desiderata

- *Representation* - simulator should cover a “universal space” of mechanisms
- *Conservative* - “margin of safety” for imperfections (c.f. Clark’s “007 Principle”)
- *Efficient* - simulator should be (much) faster than reality
- *Buildable -* results should be convertible from simulator to real robot

Evolution

- Complexity of evolving a robot increases exponentially with the complexity of the robot.
- Impractical to evolve into very large robots.
- Solution: Subsumption architectures
  - Complicated intelligent behaviors broken into simple behavior modules
  - Organized in layers which implement a particular goal
  - Higher layer subsumes the goals of layers below
  - Drawback: Whole is not always just the sum of its parts

Generativity via L-Systems



A simple rule-based system inspired by plant growth

Start with a variable (symbol), then replace it by other variables and numbers, and repeat for new symbols

L-Systems Example #1

variables : A, B
constants : none
start : A
rules : (A → ABA), (B → BBB)

http://en.wikipedia.org/wiki/L-system

L-Systems Example #1

variables : A, B
constants : none
start : A
rules : (A → ABA), (B → BBB)

A

http://en.wikipedia.org/wiki/L-system

L-Systems Example #1

variables : A, B
constants : none
start : A
rules : (A → ABA), (B → BBB)

A
ABA

http://en.wikipedia.org/wiki/L-system

L-Systems Example #1

variables : A, B
constants : none
start : A
rules : (A → ABA), (B → BBB)

A
ABA
ABABABA

http://en.wikipedia.org/wiki/L-system

L-Systems Example #1

variables : A, B
constants : none
start : A
rules : (A → ABA), (B → BBB)

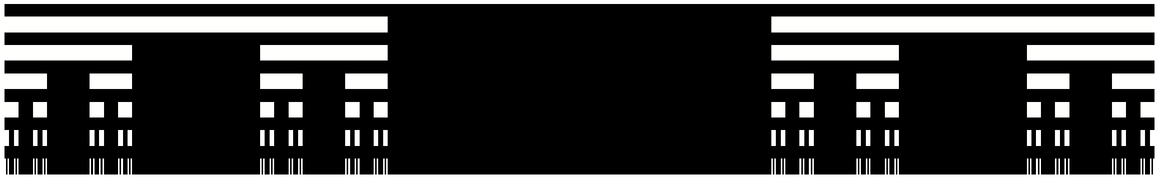
A
ABA
ABABABA
etc.

http://en.wikipedia.org/wiki/L-system

L-Systems Example #1

variables : A, B
constants : none
start : A
rules : (A → ABA), (B → BBB)

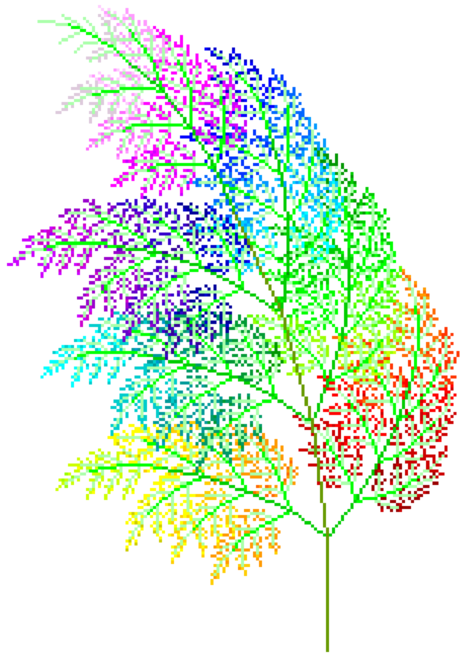
Let A mean "draw forward" and B mean "move forward".



http://en.wikipedia.org/wiki/L-system

L-Systems Example #2

variables : F
constants : 4, 5, 6, 7
start : F
rule : F → |[5+F][7-F]-|[4+F][6-F]-|[3+F][5-F]-|F



http://www.cs.unm.edu/~joel/PaperFoldingFractal/

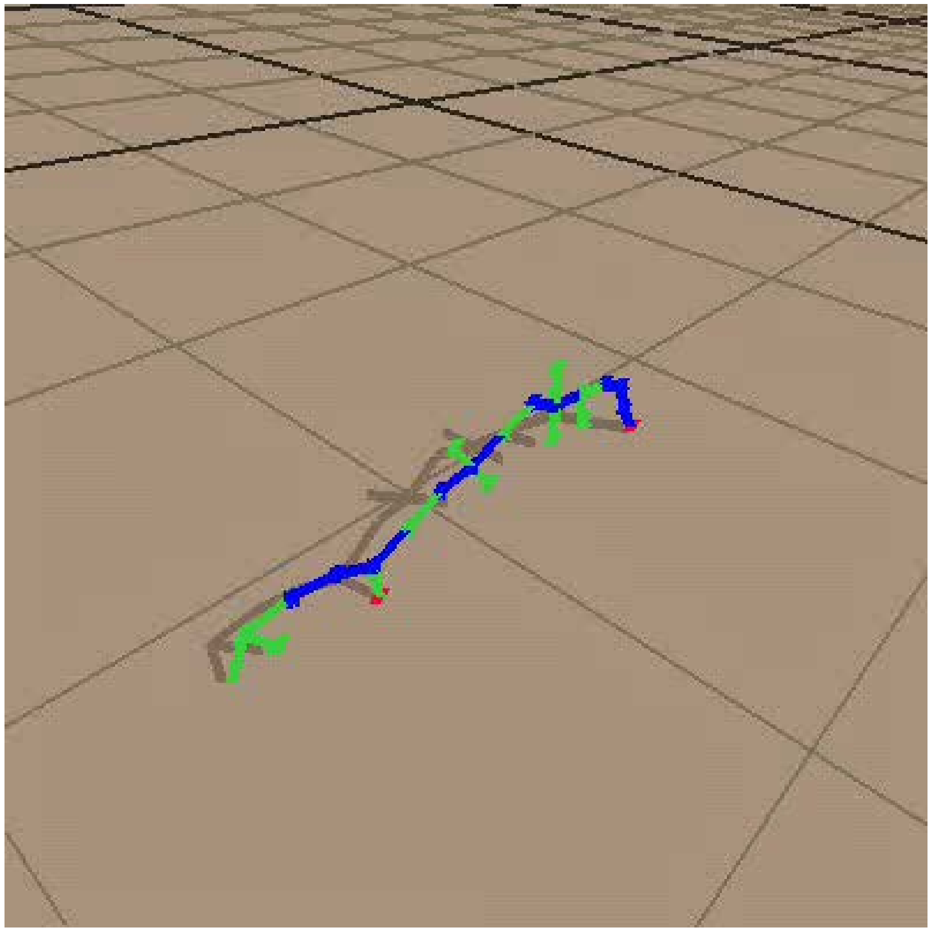
Evolving L-Systems to Build Robots (Hornby 2001)

“Genome” is L-system rules represented as sequences

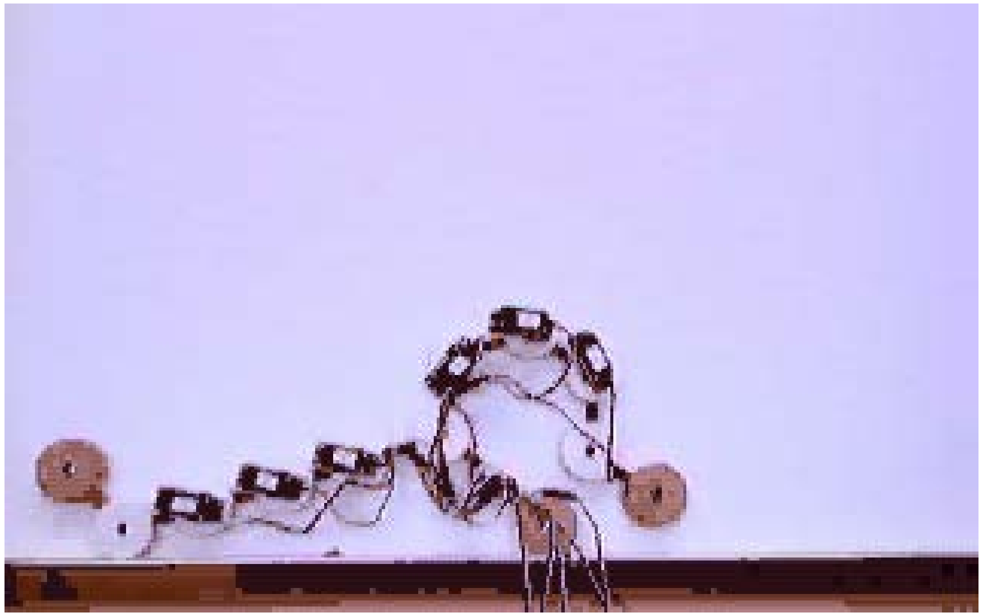
Can then crossover and mutate rules; e.g., (A $\rightarrow$ ABA) could mutate to (A $\rightarrow$ ABB), etc.

Symbols (A, B) are treated as instructions for building robot components

Evolving L-Systems to Build Robots (Hornby 2001)

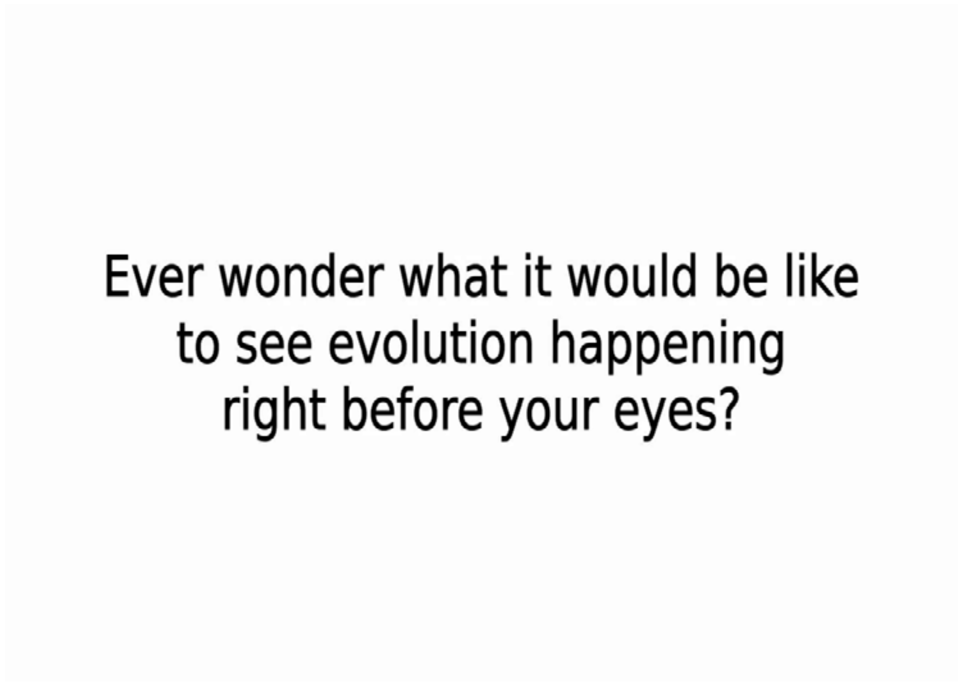


Evolving L-Systems to Build Robots (Hornby 2001)

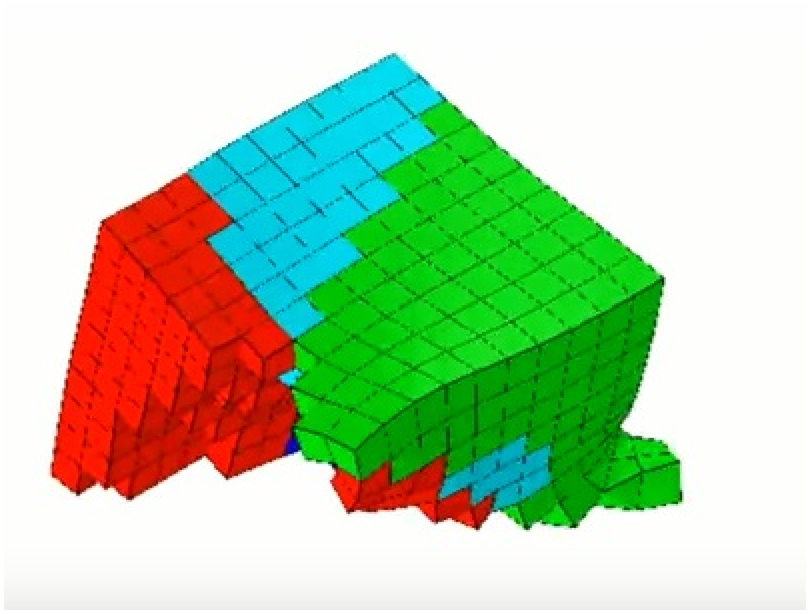


https://www.youtube.com/watch?v=W4wAubXVraA

Evolving Soft Robots (Lipson 2013)



Ever wonder what it would be like to see evolution happening right before your eyes?



https://www.youtube.com/watch?v=z9ptOeByLA4

Challenges

- Current ER systems are often very simple and could be easily bettered by hand-coded solutions.
- Many control-system evaluations achieve satisfactory results in unacceptable runtimes.
- The robot may also enter into dangerous states in which its hardware could be damaged.
- Many unanswered questions drive this as an area of research.

Approaching the challenges

Evolutionary Robots: Let them be babies first

- Like tadpoles become frogs, change their body forms while learning how to walk, Robots have been evolved, spending less time in “infant” tadpole-like forms and more time in “adult” four-legged forms.

Conclusion

- Evolutionary algorithms are used to generate control algorithms using the Darwinian principle of survival-of-the-fittest.
- Most of the work has focused on the development of control systems for autonomous mobile robots. Some researchers have used the techniques to evolve robotic hardware configurations.

Evolutionary Robotics

THE END

Questions?