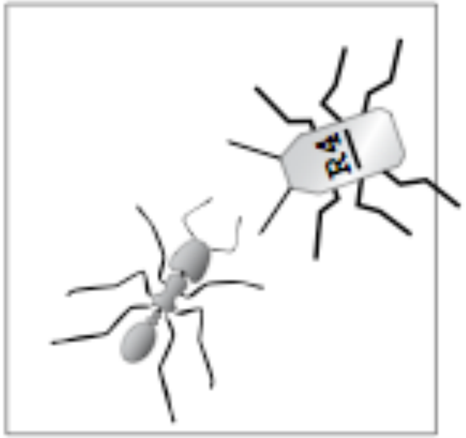
Evolutionary Robotics



A definition

Evolutionary Robotics is automatic generation of robot control systems and morphologies by means of artificial evolution.

The control systems are often neural networks.

<u>Two drivers</u>:

- **Engineering**: a tool to investigate the space of possible control strategies and body design for autonomous robots

- **Biology**: A *synthetic* (as opposed to an *analytic*) approach to the study of the mechanisms of adaptive behavior in machines and animals (Braitenberg, 1984)

Evolution of Neural Networks

Genotype can encode:

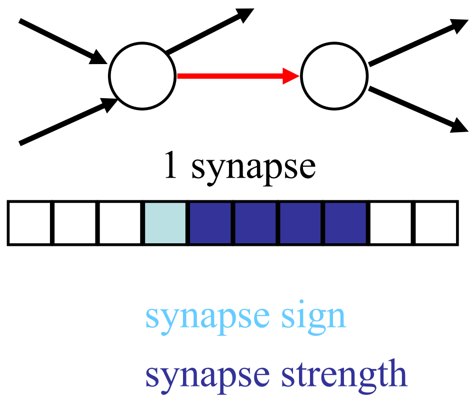
1. Connection Weights: pre-defined network architecture, each weight encoded in separate genes (binary or real-valued), fixed-length genotype

2. Topology: variable-length genotype encodes presence/type of neurons and their connectivity

3. Learning Rules: fixed or variable length genotype encodes learning rules (constants of polynomial expression of x and y), not weights.

Evolution of Weights

Use real-valued or binary representation
Each synaptic weight is represented by one or more genes; e.g.:

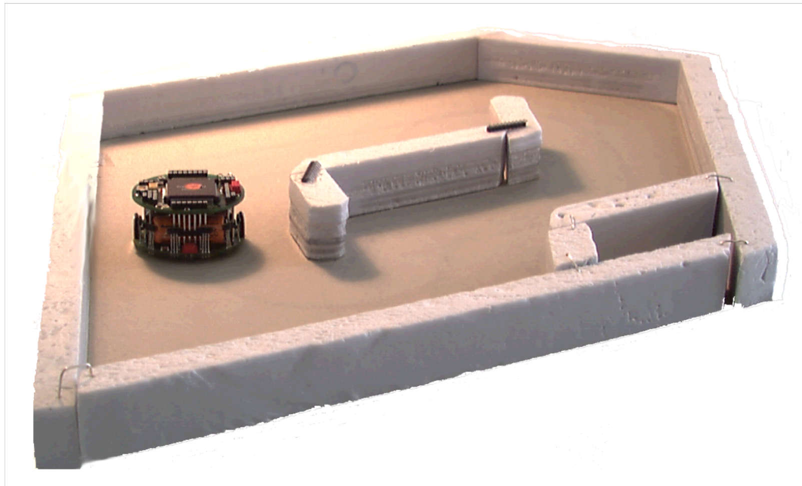


Fitness function can be error, as in Back-Prop, or higher-level consequence of network output, such as behavior of a robot

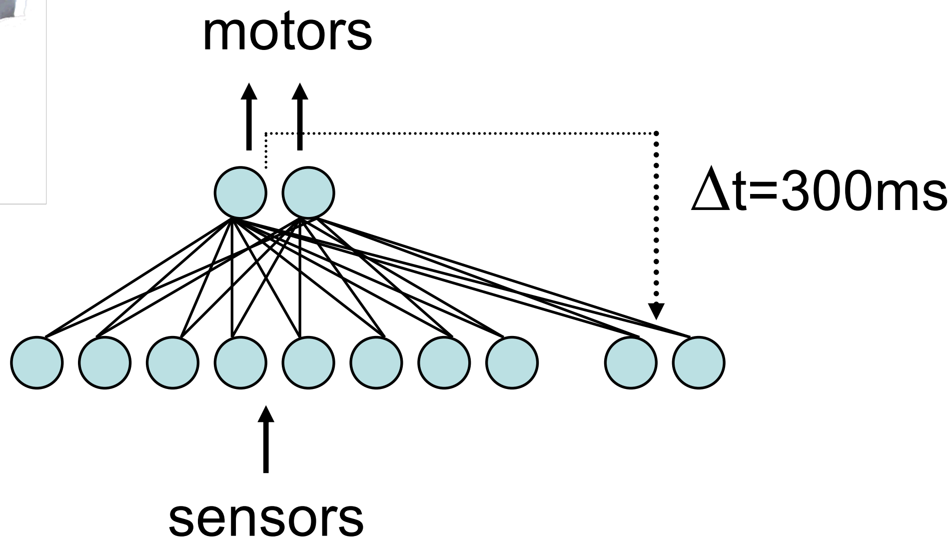
Can be combined with learning:

- learning starts from genetically encoded weights
- fitness measures performance of network after training
- learned weights are not written back into genome

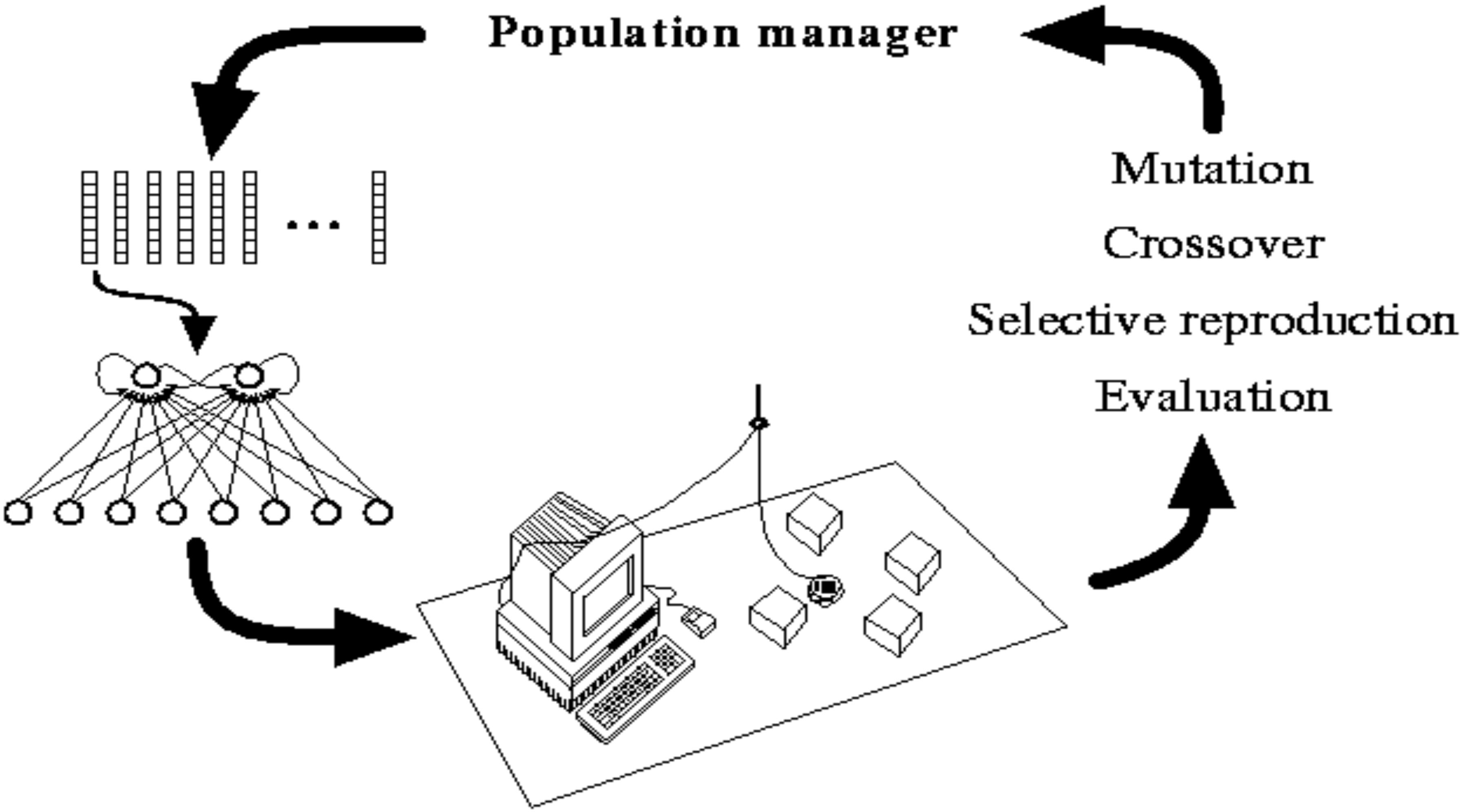
Collision-free Navigation



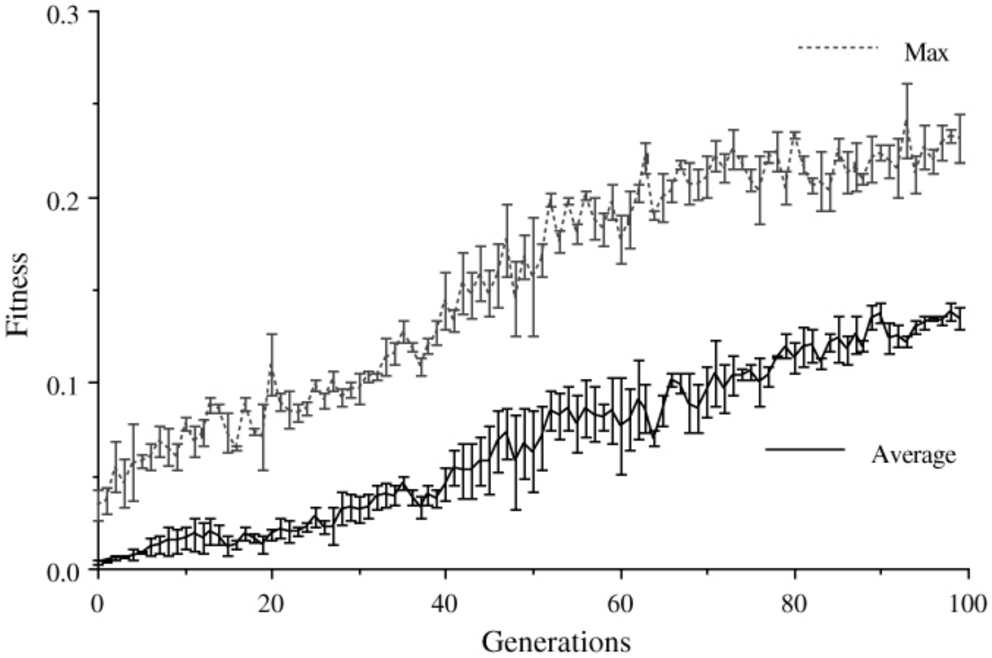
$$\text{Fitness} = V \times \Delta v \times (1-s)$$

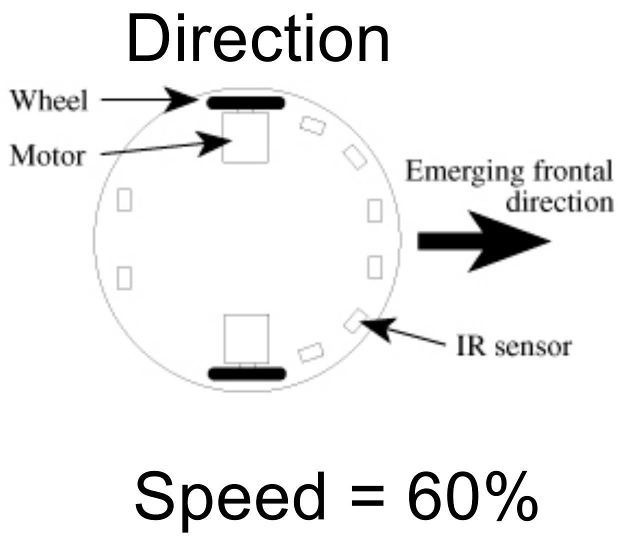


Methodology



Results



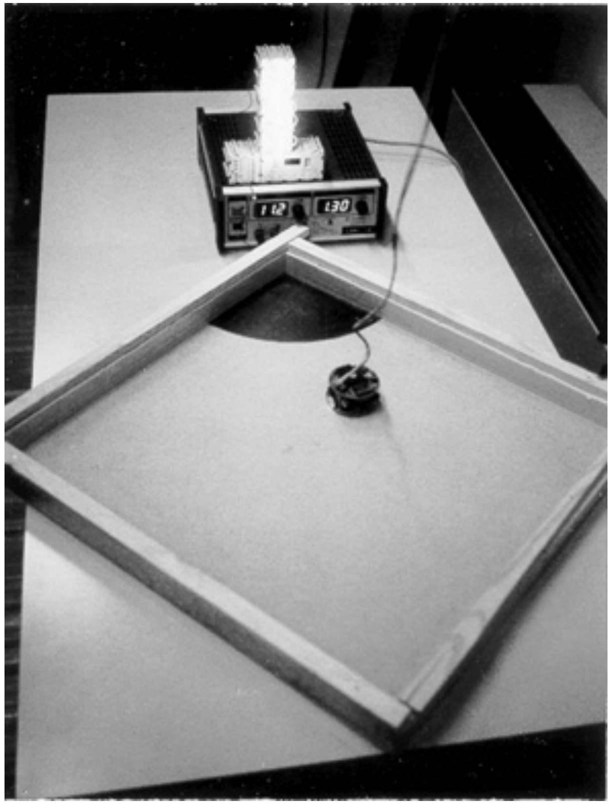


The average and best population fitness are typical measures of performance.

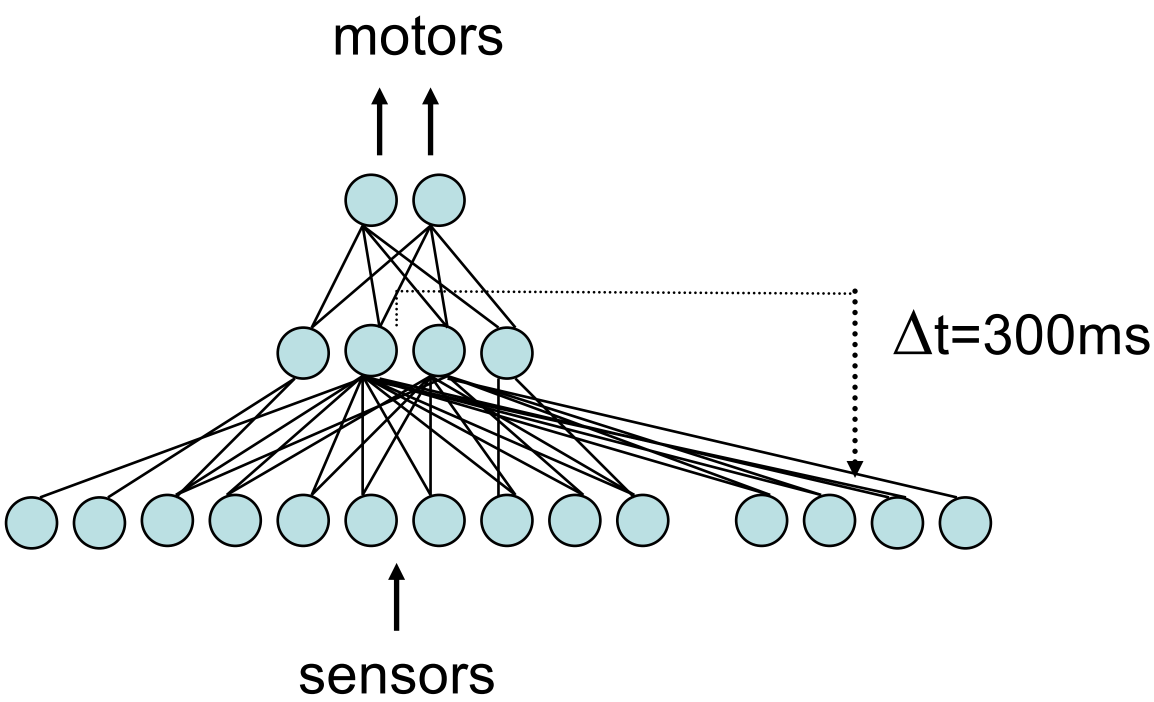
Evolved robots always have a preferential direction of motion and speed.

Homing for Battery Charge

Let us now put the robot in a more complex environment and make the fitness function even simpler. The robot is equipped with a battery that lasts only 20 s and there is a battery charger in the arena.

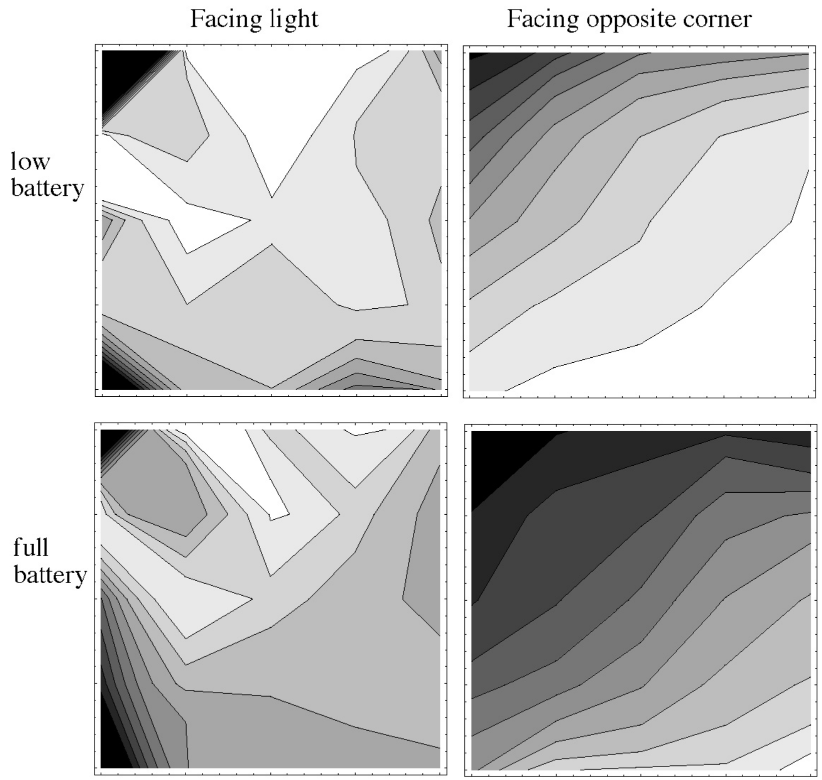


$$\text{Fitness} = V \times (1\text{-}s)$$

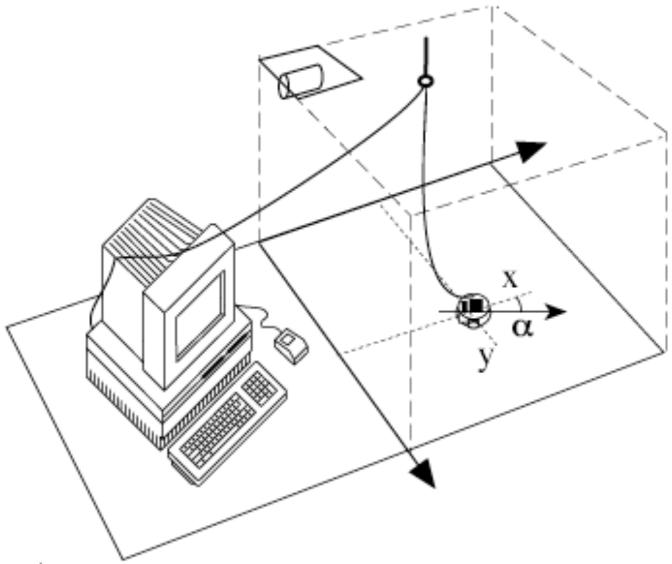


Machine Neuro-Ethology

After 240 generations, we find a robot capable of moving around and going to recharge 2 seconds before the batteries are completely discharged.

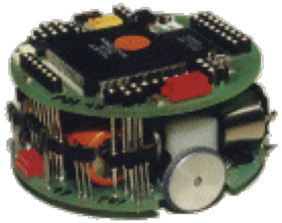


Neural Activity Maps



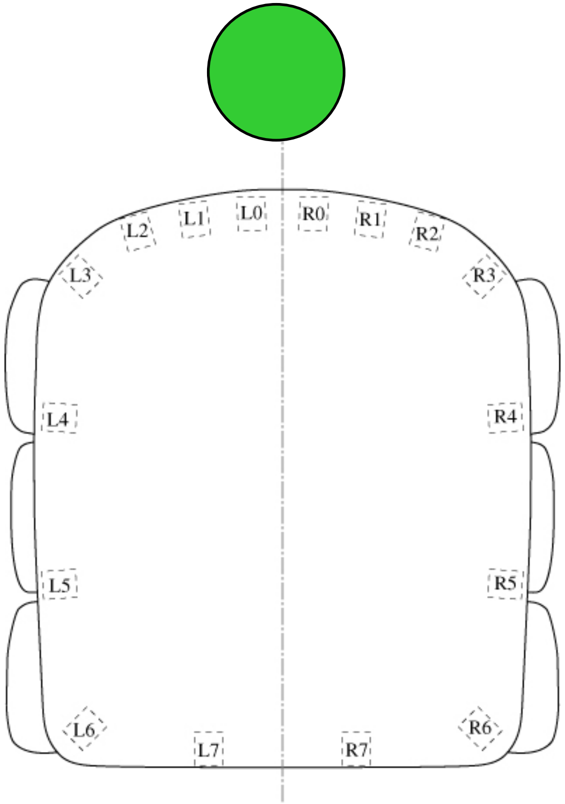
Evolution of complex robots

It is difficult to evolve from scratch large and complex robots because of:
- hardware robustness
- *bootstrap problem*



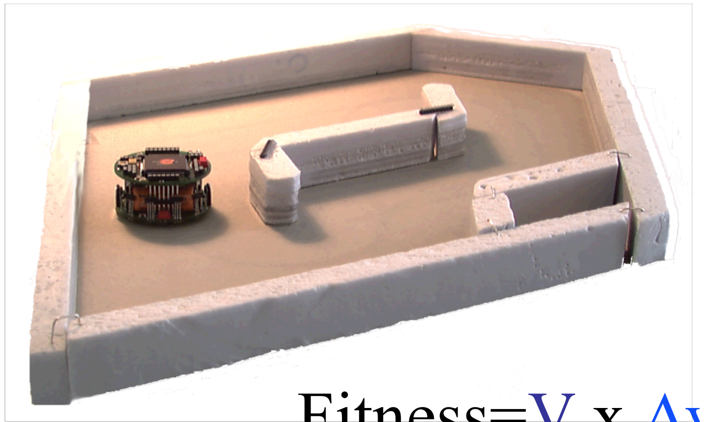
Khepera

Koala



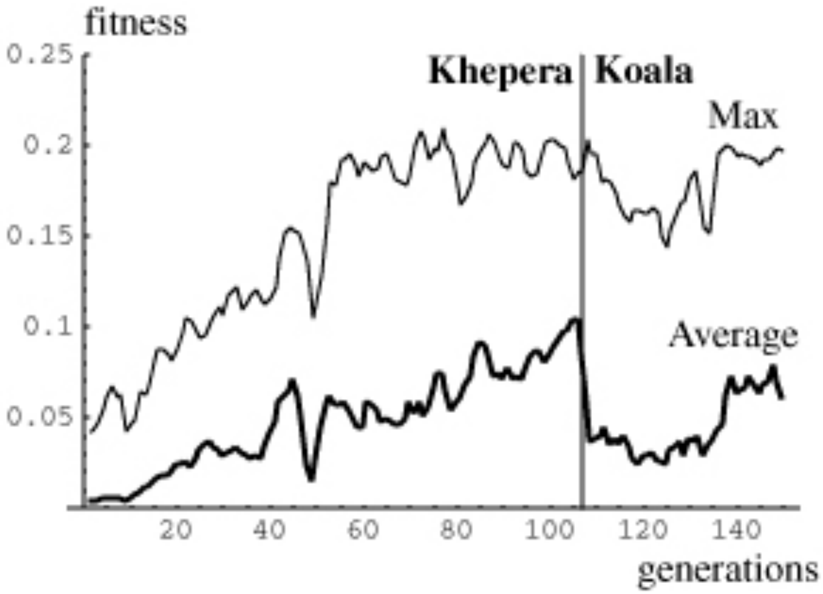
Incremental evolution

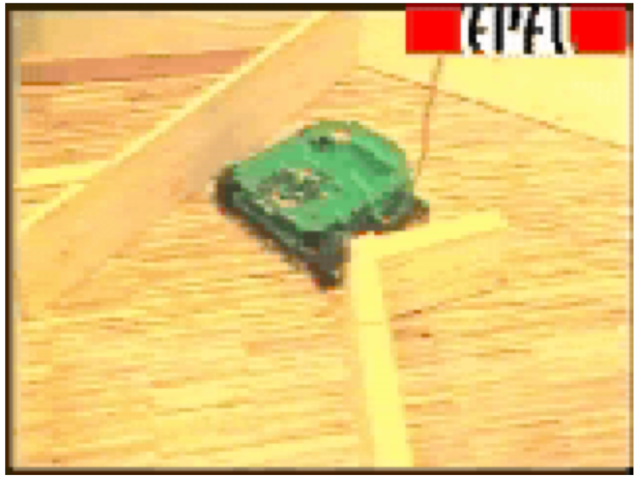
A solution is to incrementally evolve robots from simple to complex. Simple robots gradually generates solutions that can be adapted to more complex robots faster and better than by starting with a complex robot.



Fitness=V x Δv x (1-s)

Incremental evolution results



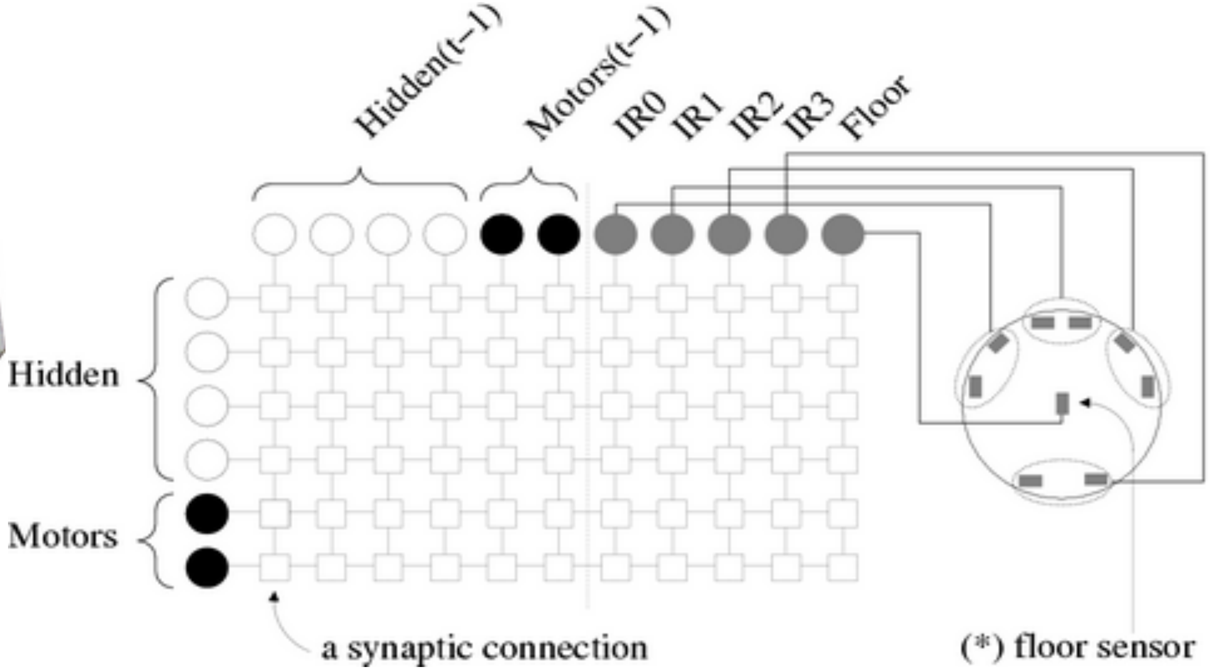


Testing Biological Hypothesis

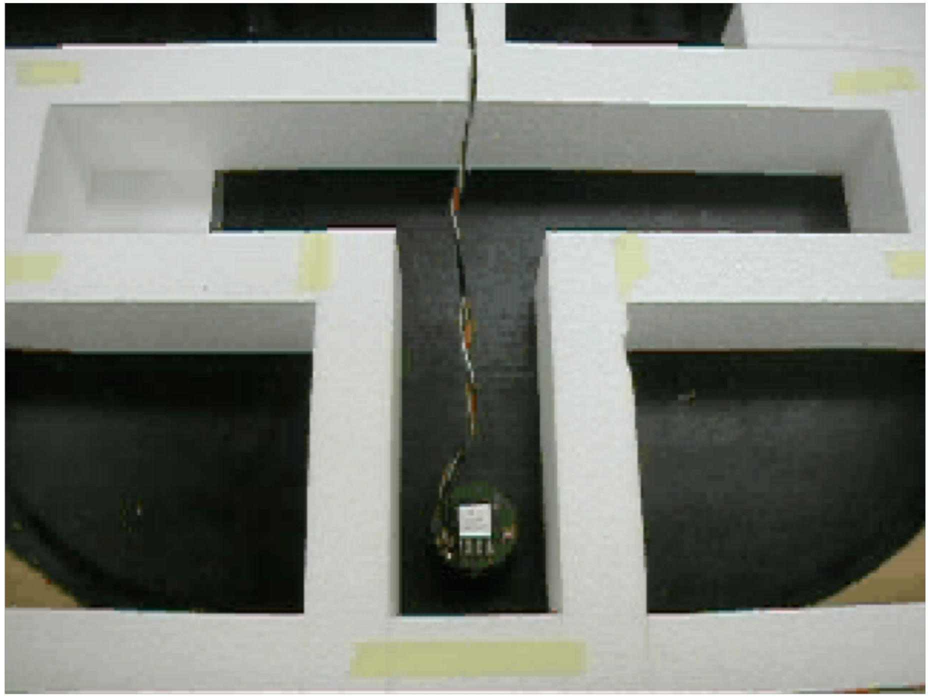
Reinforcement Learning: Since Pavlov's experiments with dogs, it has been known that positive and negative reinforcement signals are used to learn behaviors

But is synaptic modification really necessary?

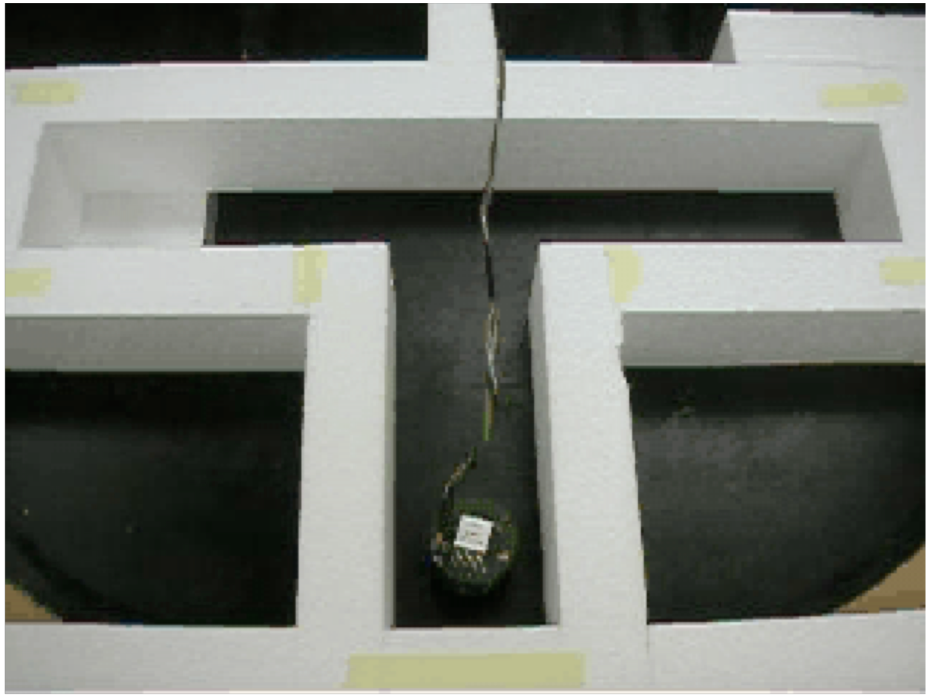
Blynel and Floreano, 2003



T-maze without synaptic modification

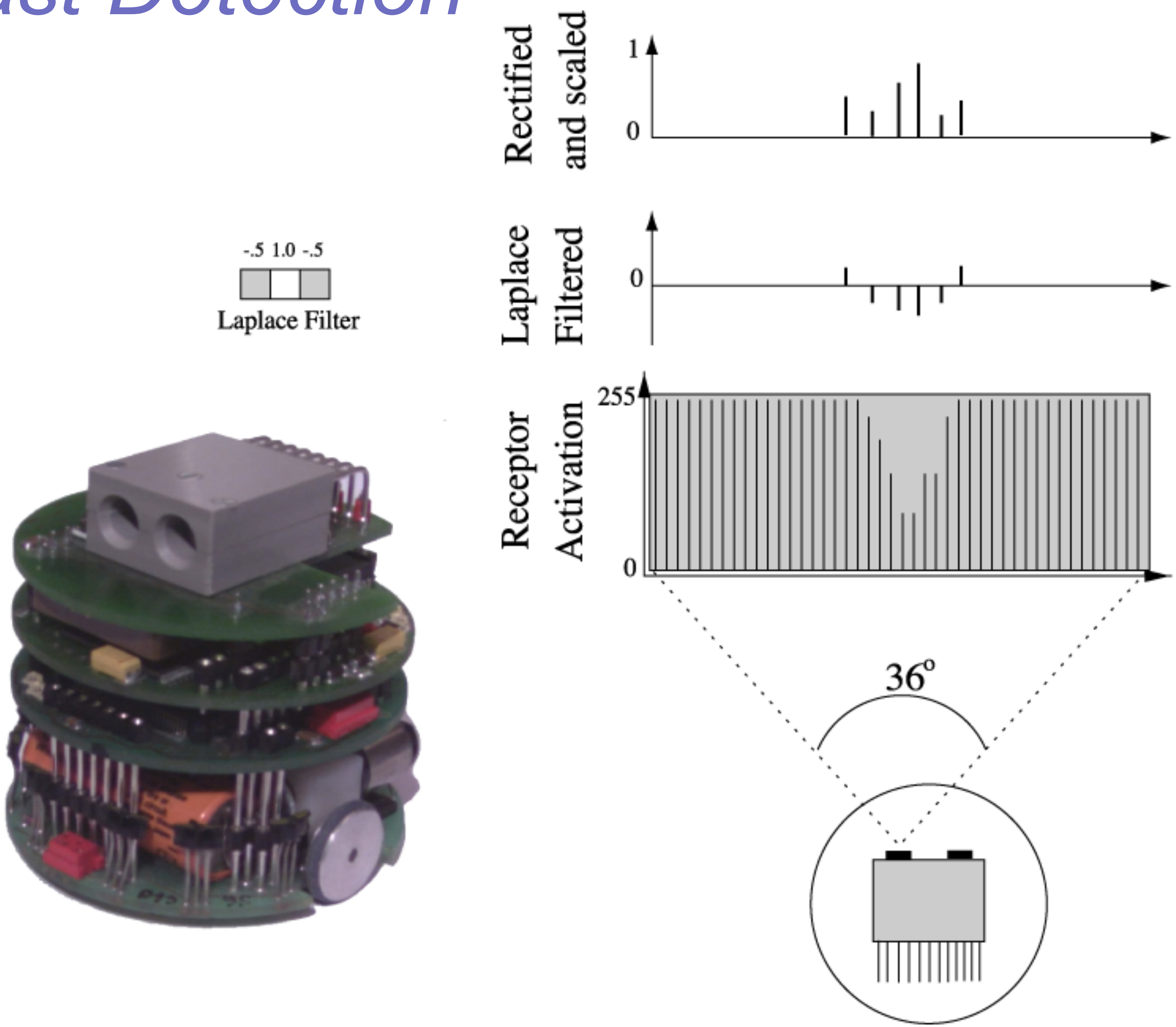


Trial 1



Trial 2

Contrast Detection



Vision-based Navigation

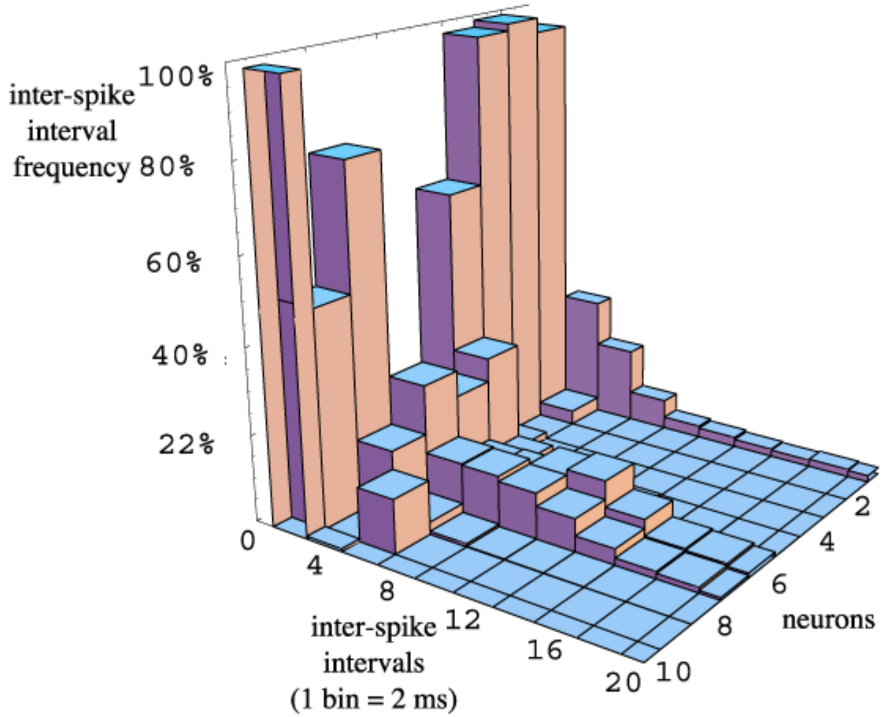
Fitness proportional to amount of forward translation over 2 mins

After 30 generations

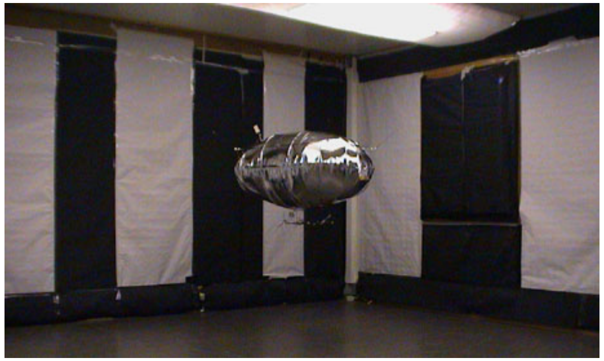
Neural Dynamics

Neuron #

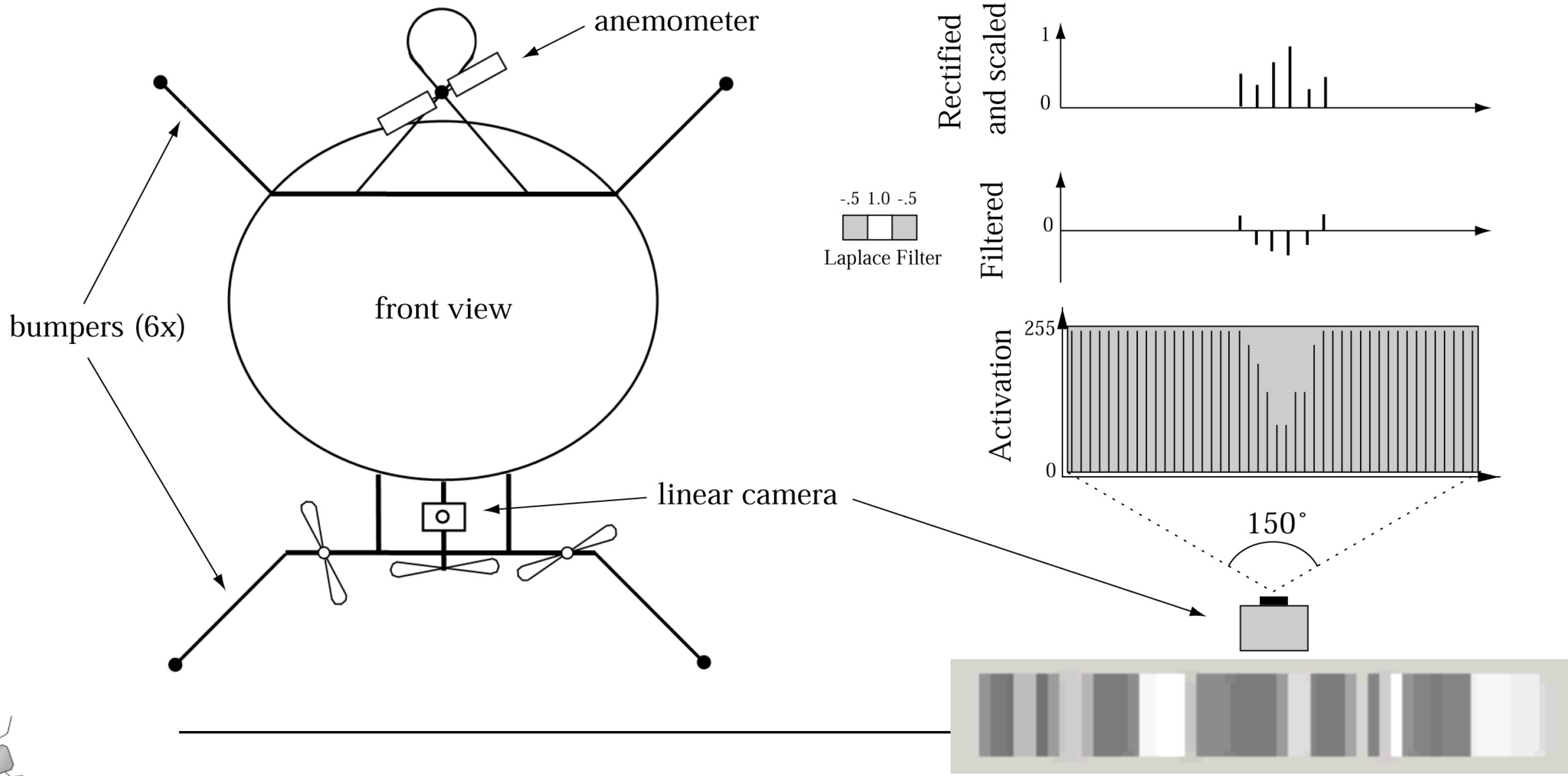
	1	2	3	4	5	6	7	8	9	10
spikes/s	9	445	453	450	330	40	129	363	0	452



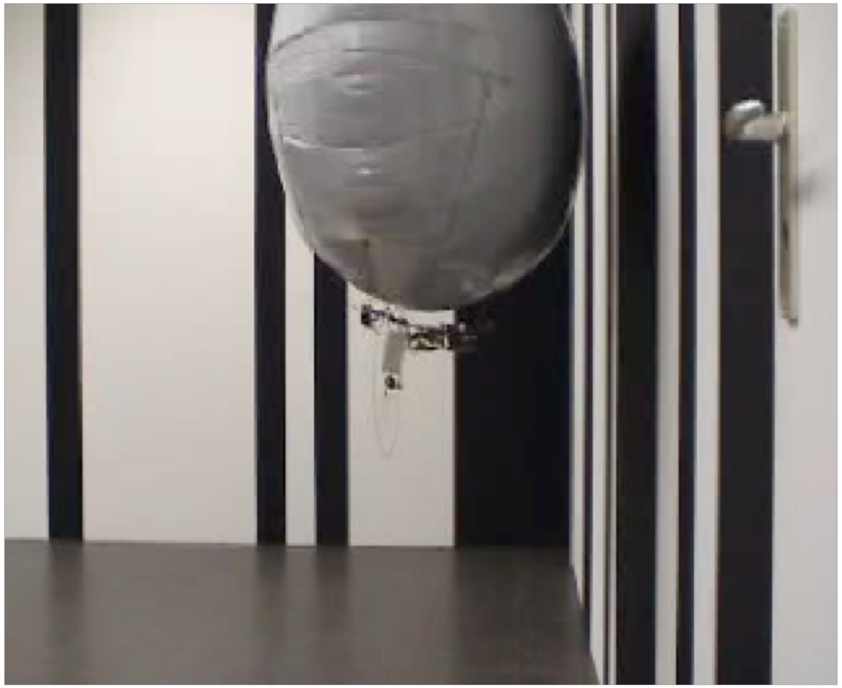
Vision-based Blimp



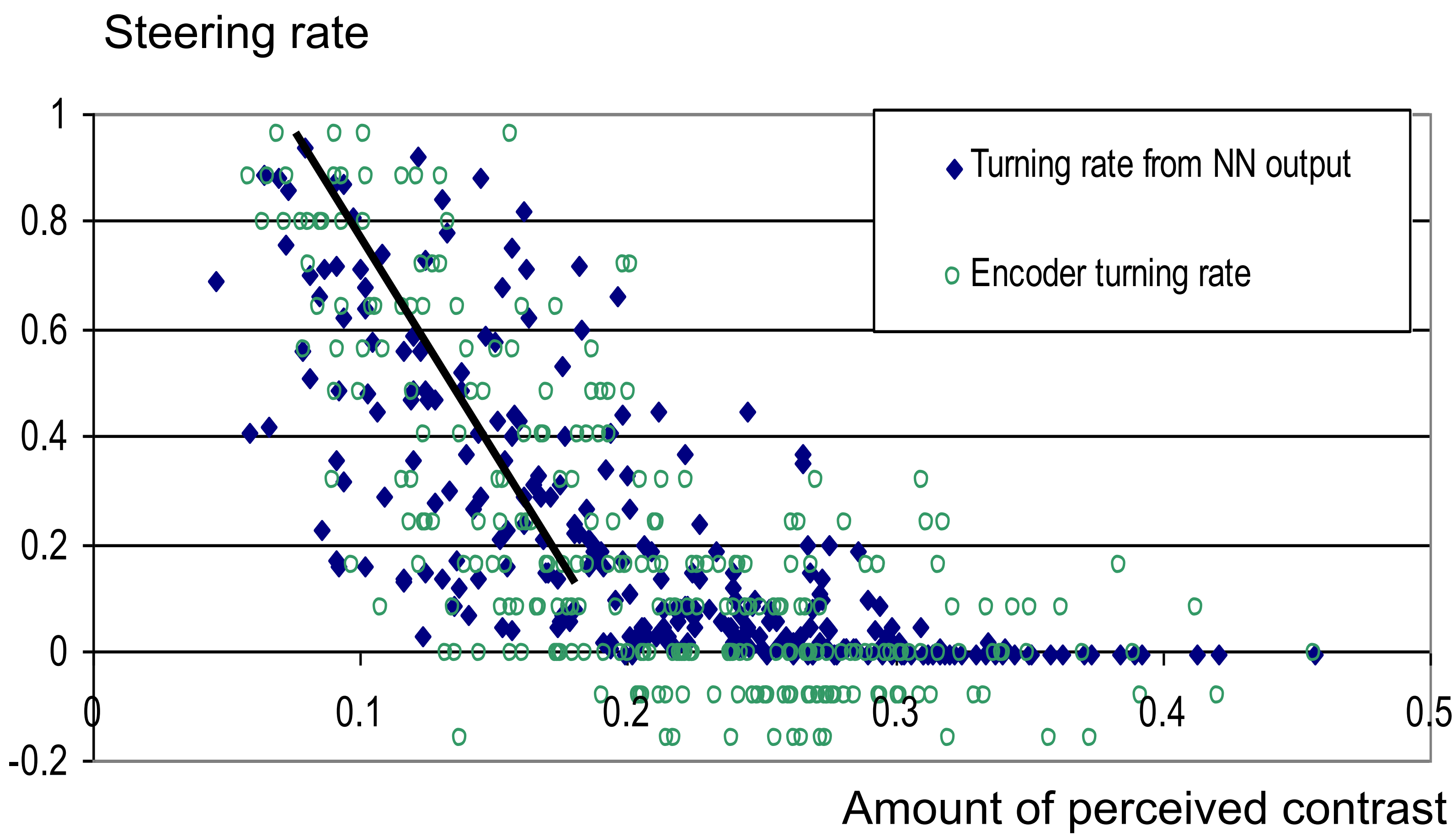
- 5 x 5 room, random size stripes
- Fitness = forward motion (anemometer)
- 2 trials, 2 minutes each
- Evolution + network activation on PC
- Sensory pre-processing on microcontroller

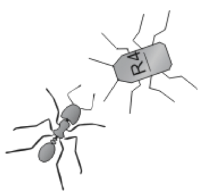


Evolved behaviour

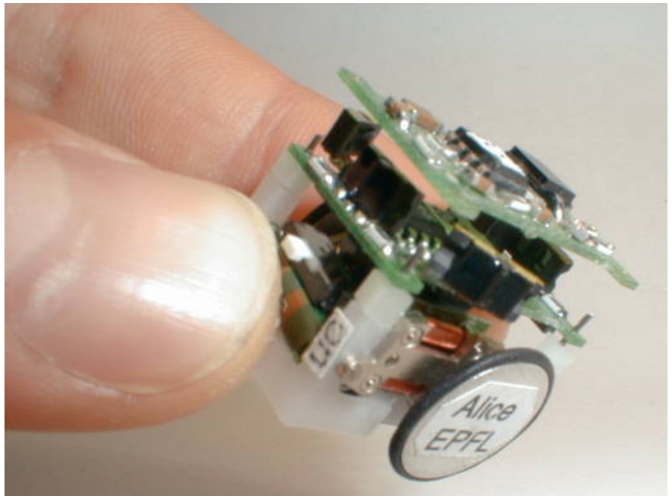



Evolved Control Strategy

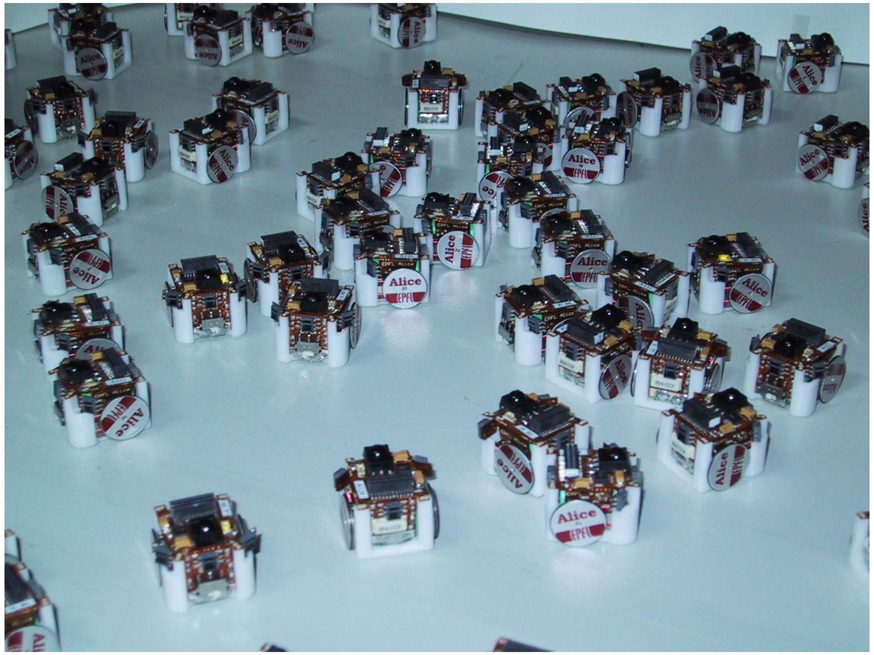


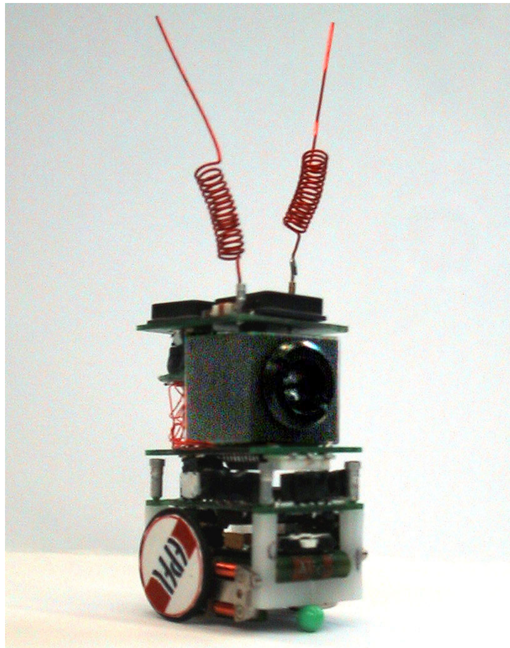


Alice Micro-Robot

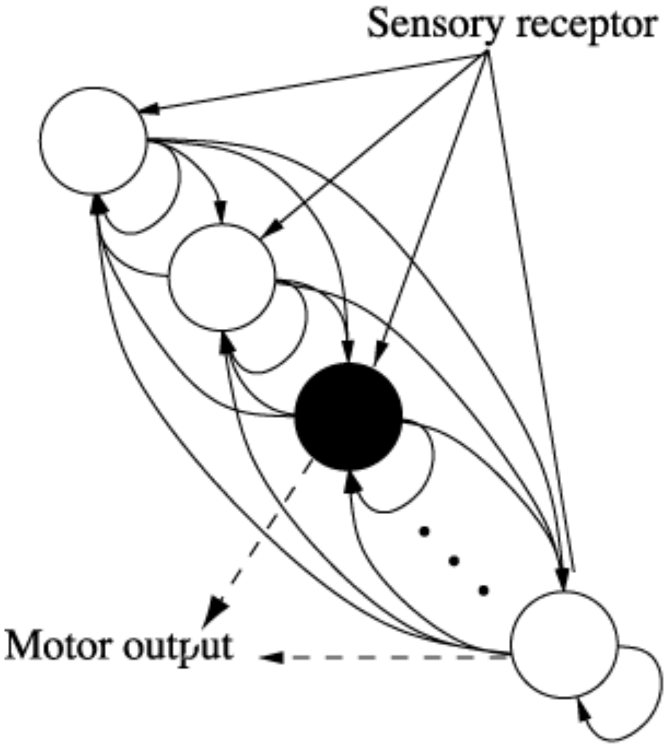


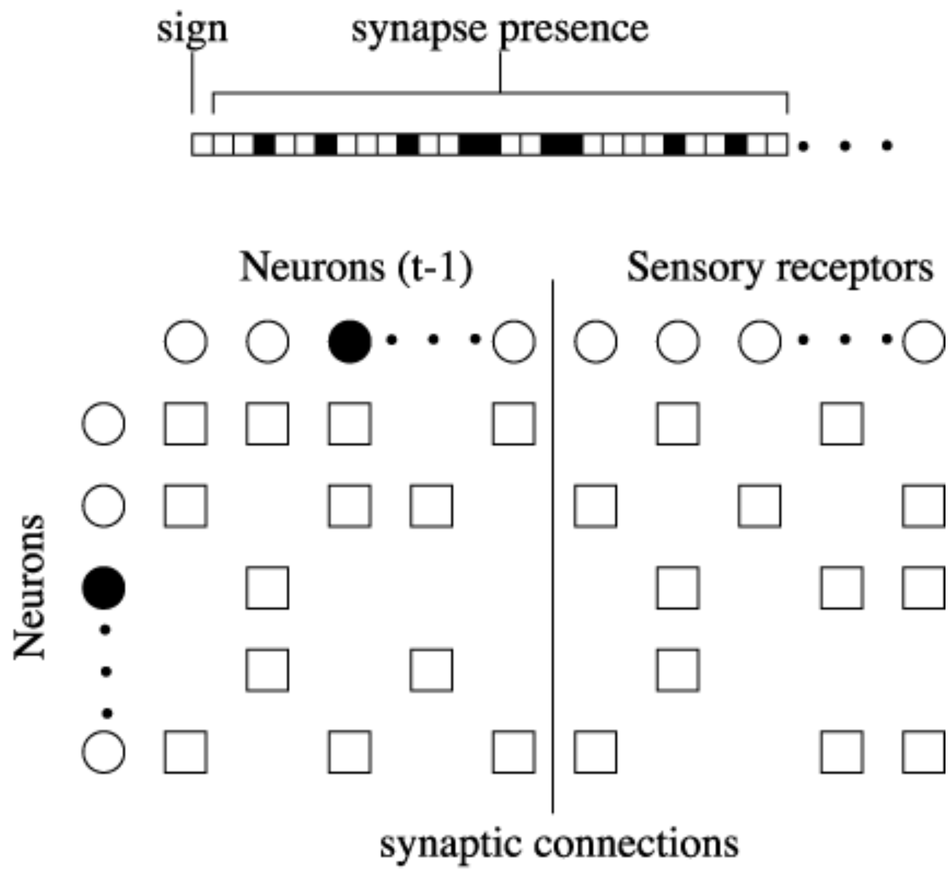
- microcontroller PIC16F84
- 2mA @ 5V
- 10 hours autonomy
- 2 swatch motors
- 4 proximity sensors
- modular (vision, radio, etc.)





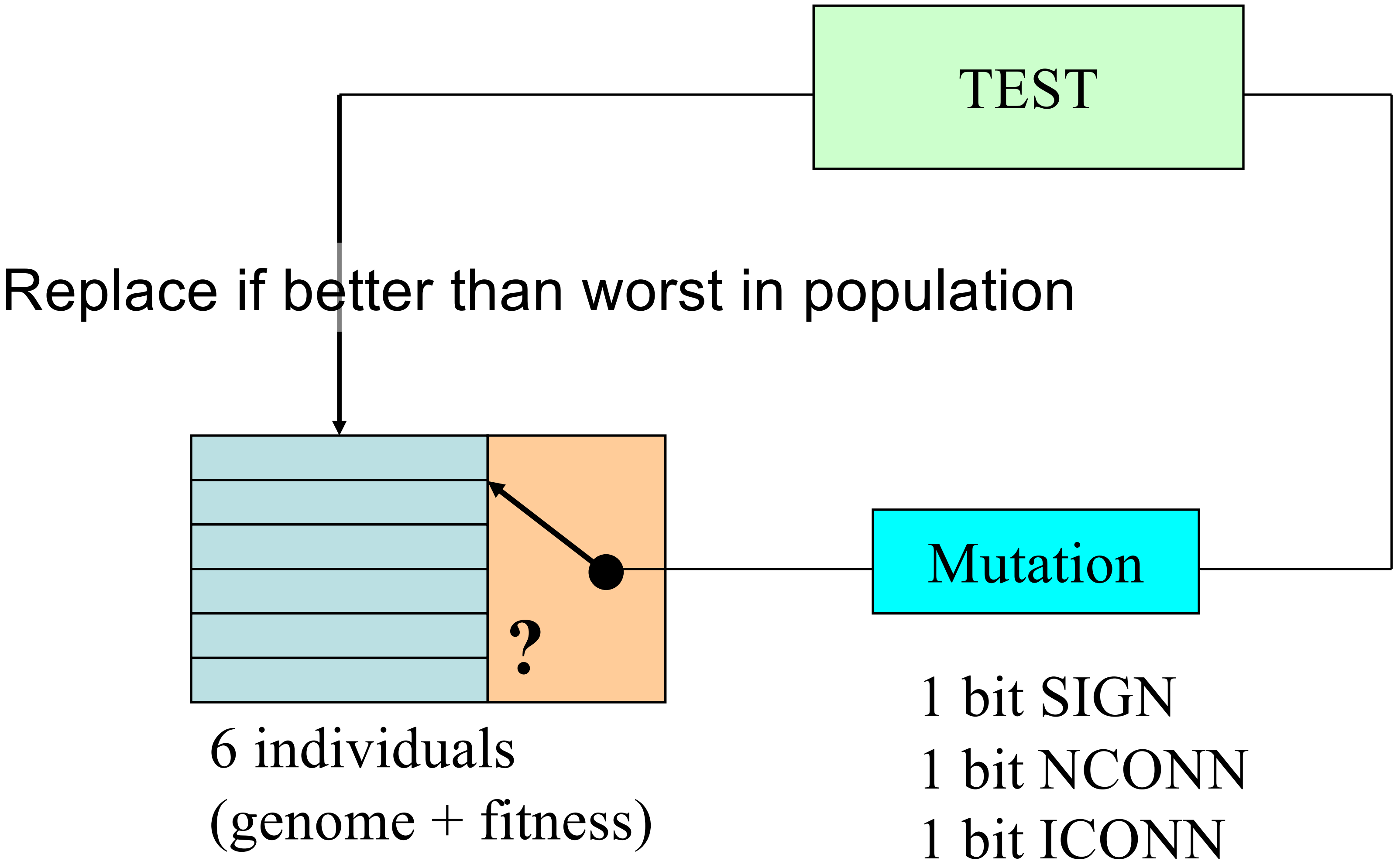
Genetic encoding





Evolution of connectivity and neuron sign only
For existing connections, strength = 1

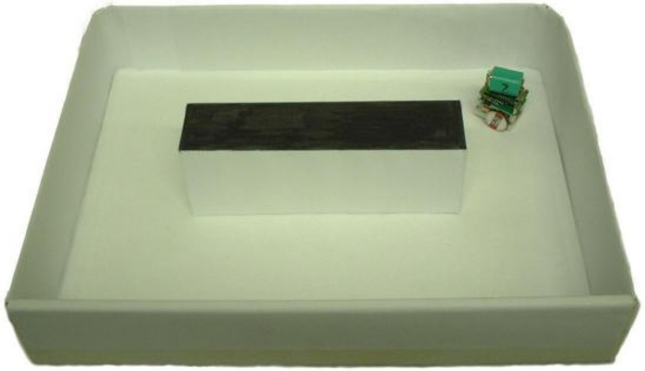
Steady-State Genetic Algorithm



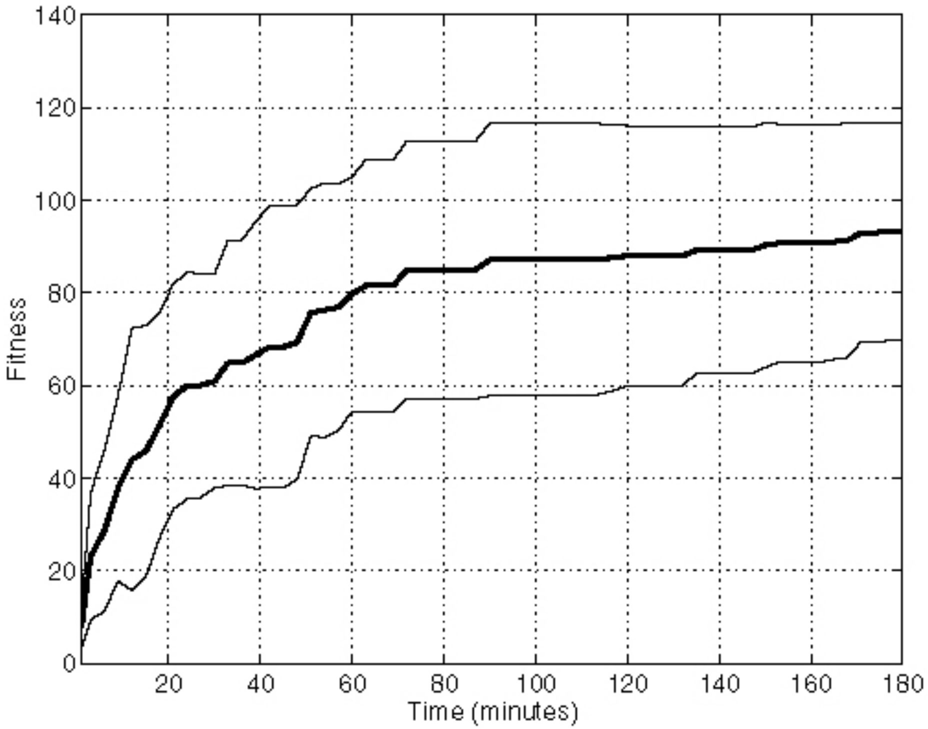
Bit-size Evolution

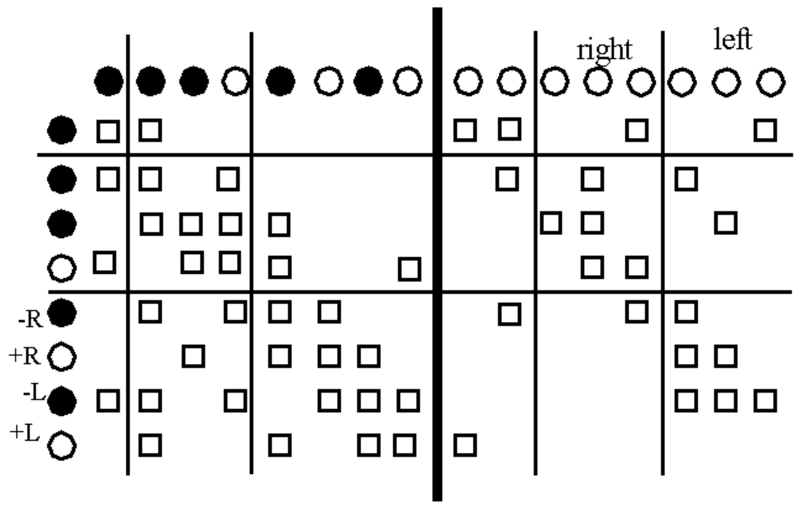
Forward navigation with obstacle avoidance

$$\text{Fitness} = V \times \Delta v \times (1-s)$$



Steady-state evolution

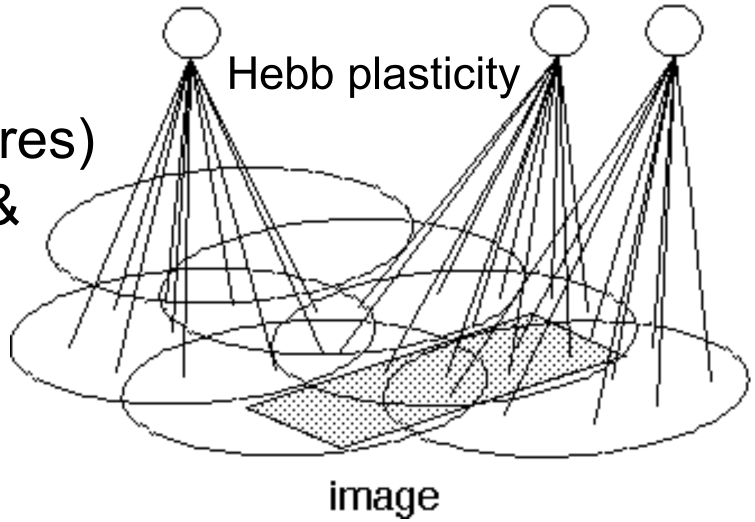




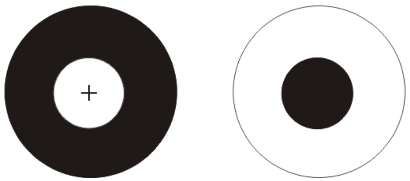
- bias: ↻
- IR Right: ↺
- IR Left: ↻

Visual features

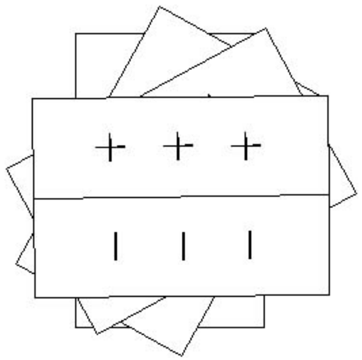
Process whereby visual neurons become sensitive to certain sensory patterns (features) during the developmental process (Hubel & Wiesel, 1959)

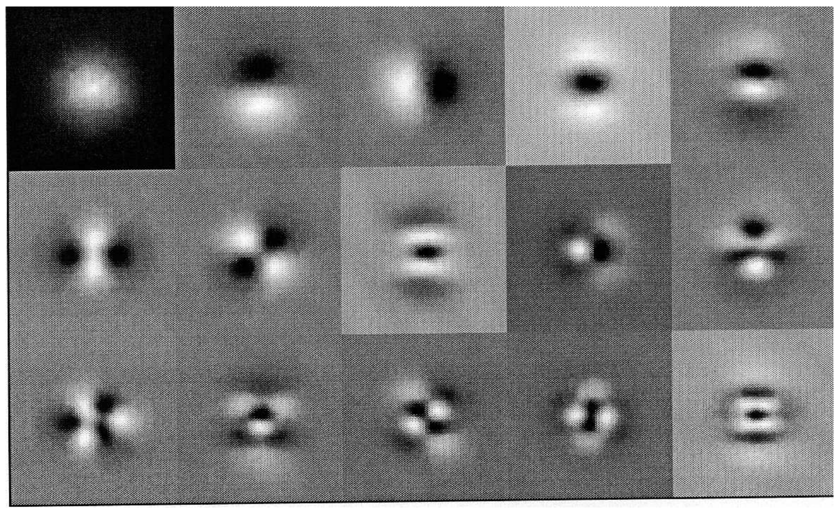


Center-Surround

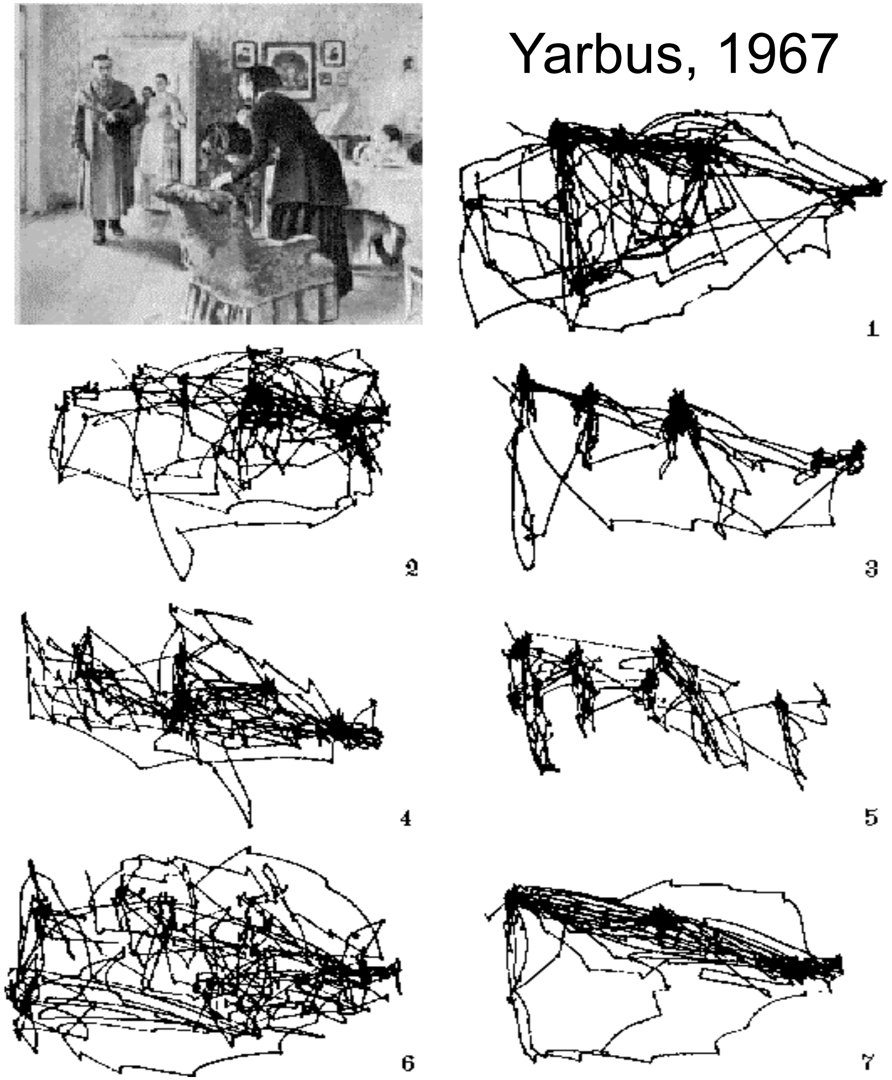


Oriented Edges



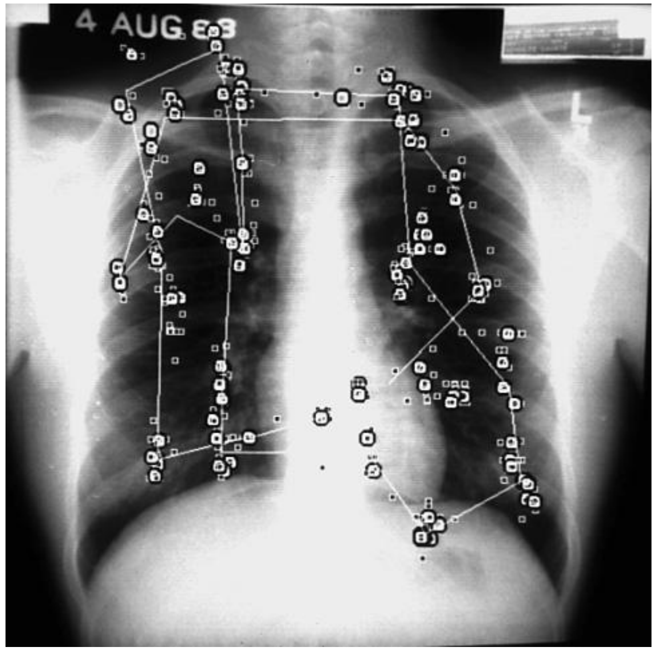


Active vision



Yarbus, 1967

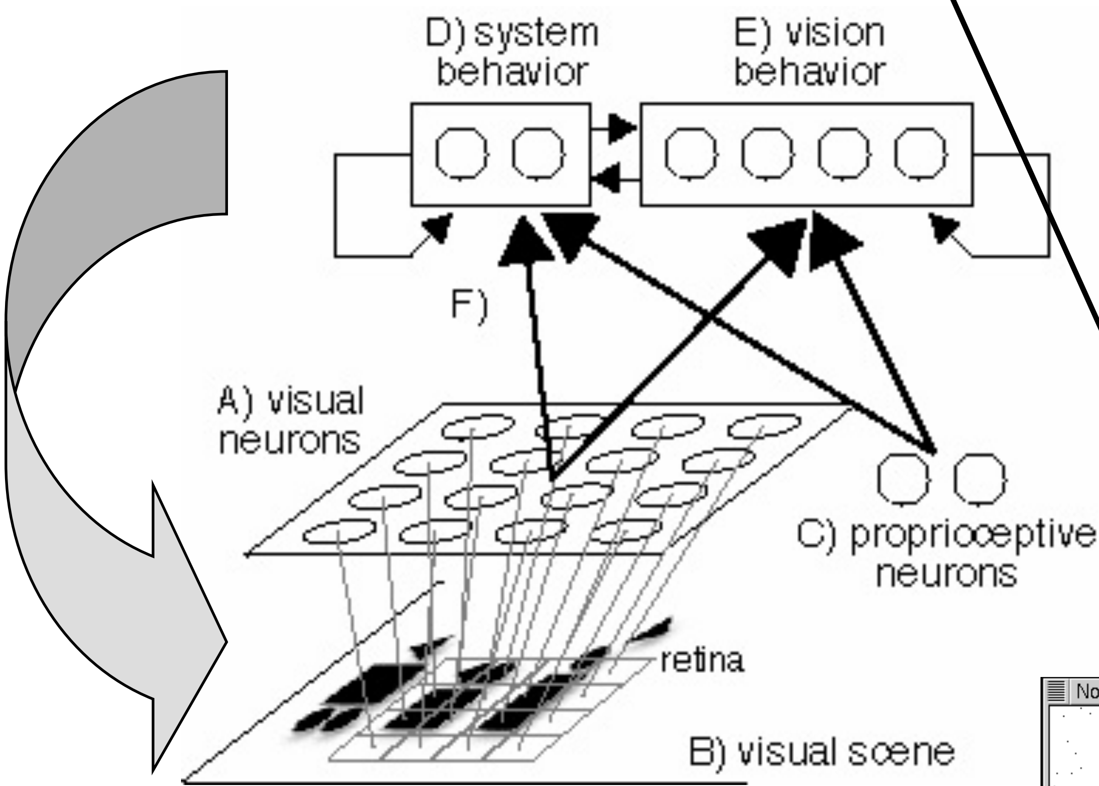
Process of selecting by motor actions sensory patterns (features) that make discrimination easier (Bajcsy, 1988)

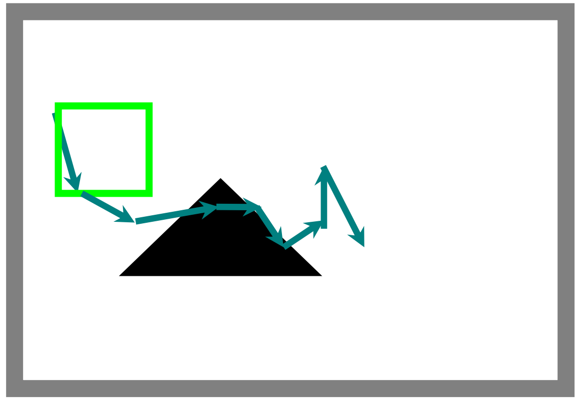


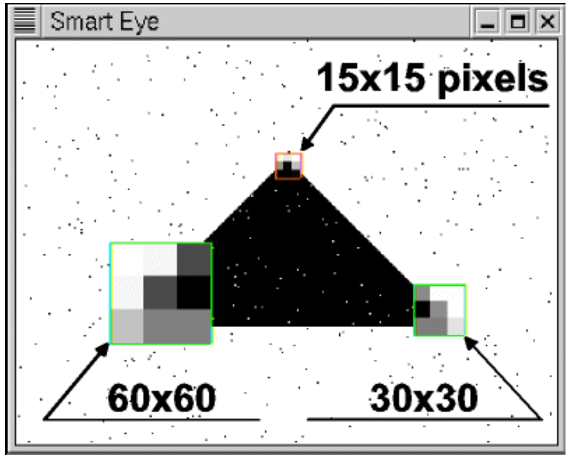
Active Vision

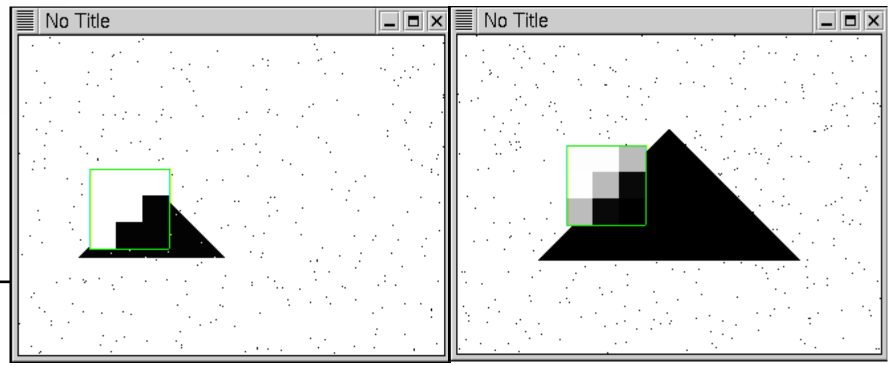
shape discrimination
robot control
car driving

retina movement
zooming factor
filter type





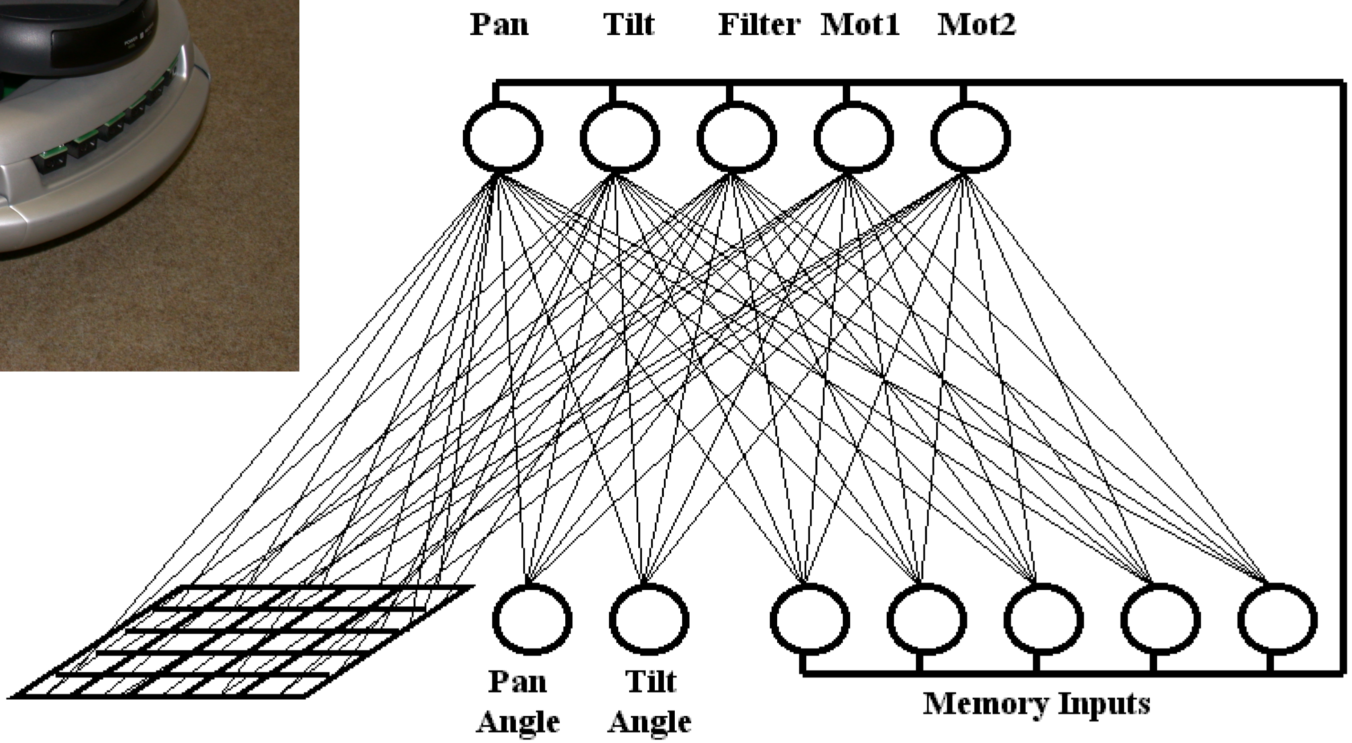




Robot Navigation

Goal: Robot must move around simple arena using only vision information from a pan/tilt camera.

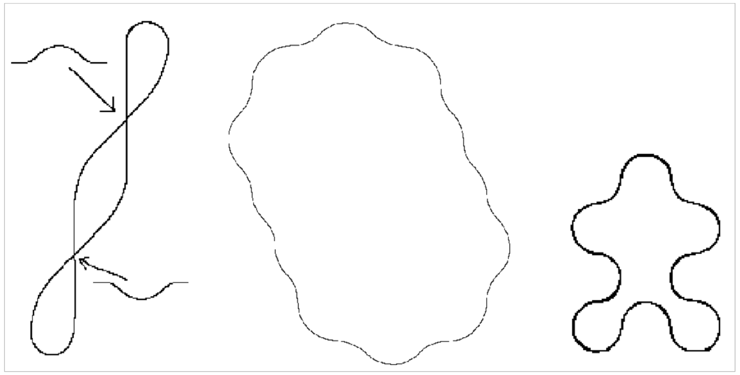
Output of vision system is movement of camera (pan/tilt) and of robot wheels (mot1/mot2). Filter as before.



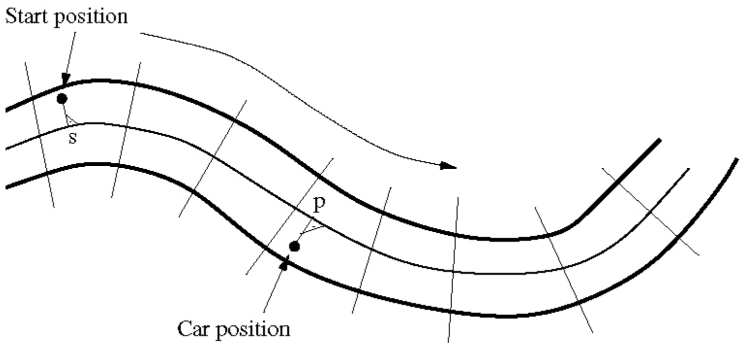

Environment

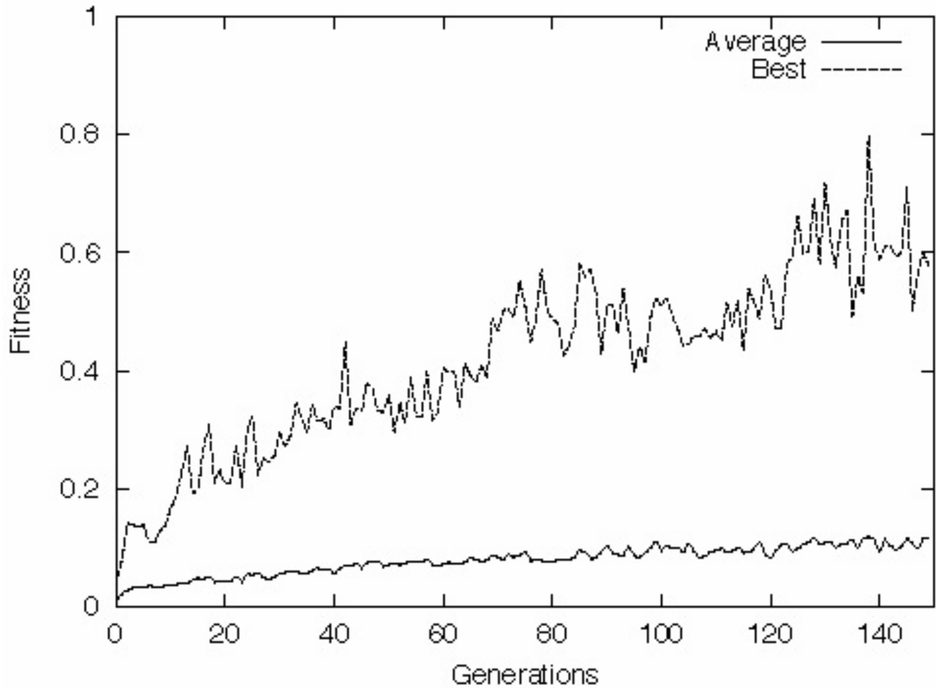
Car Driving

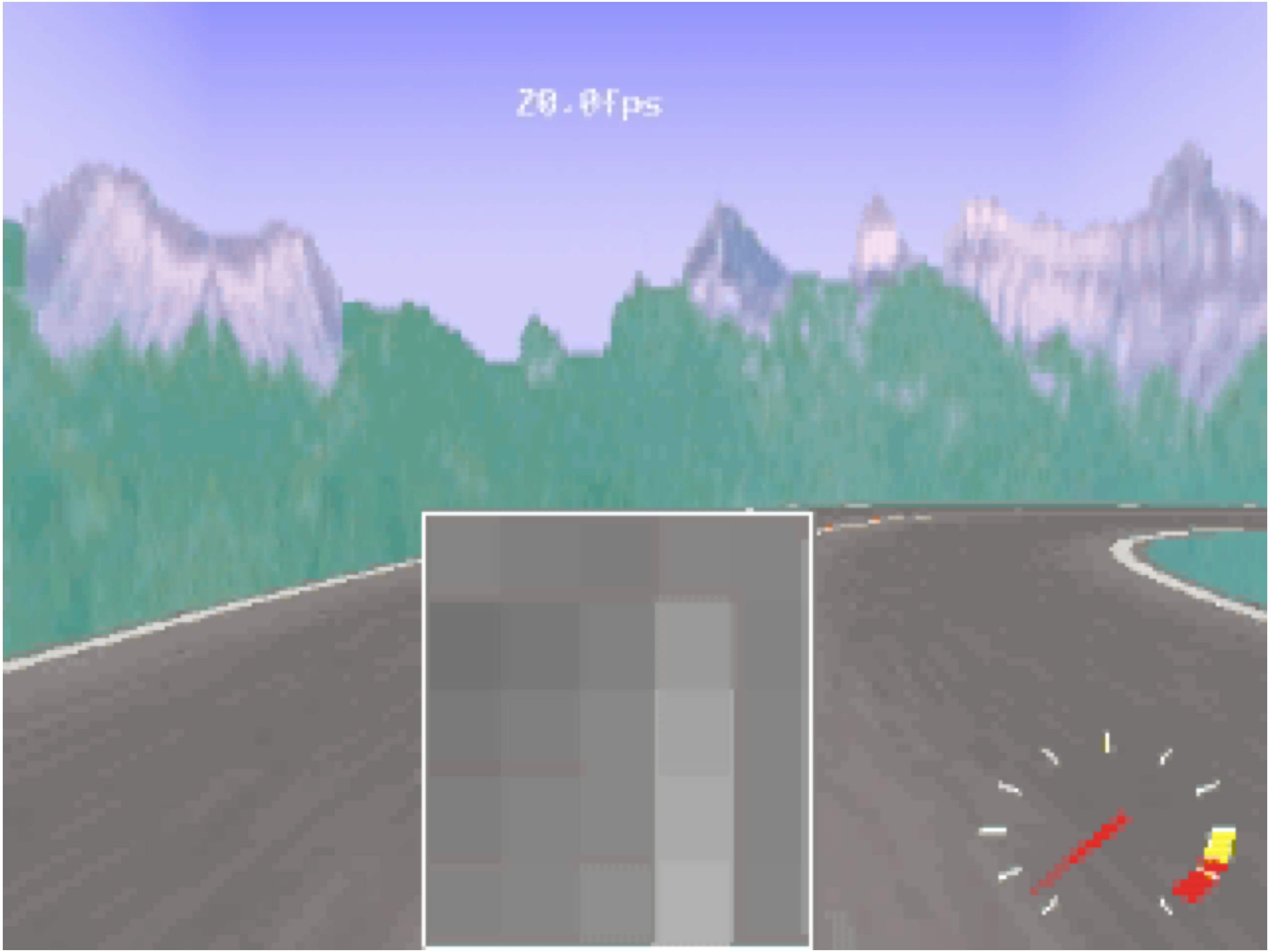
<u>Fitness</u> = percentage of covered distance D in R races on M circuits (limited time for each race).



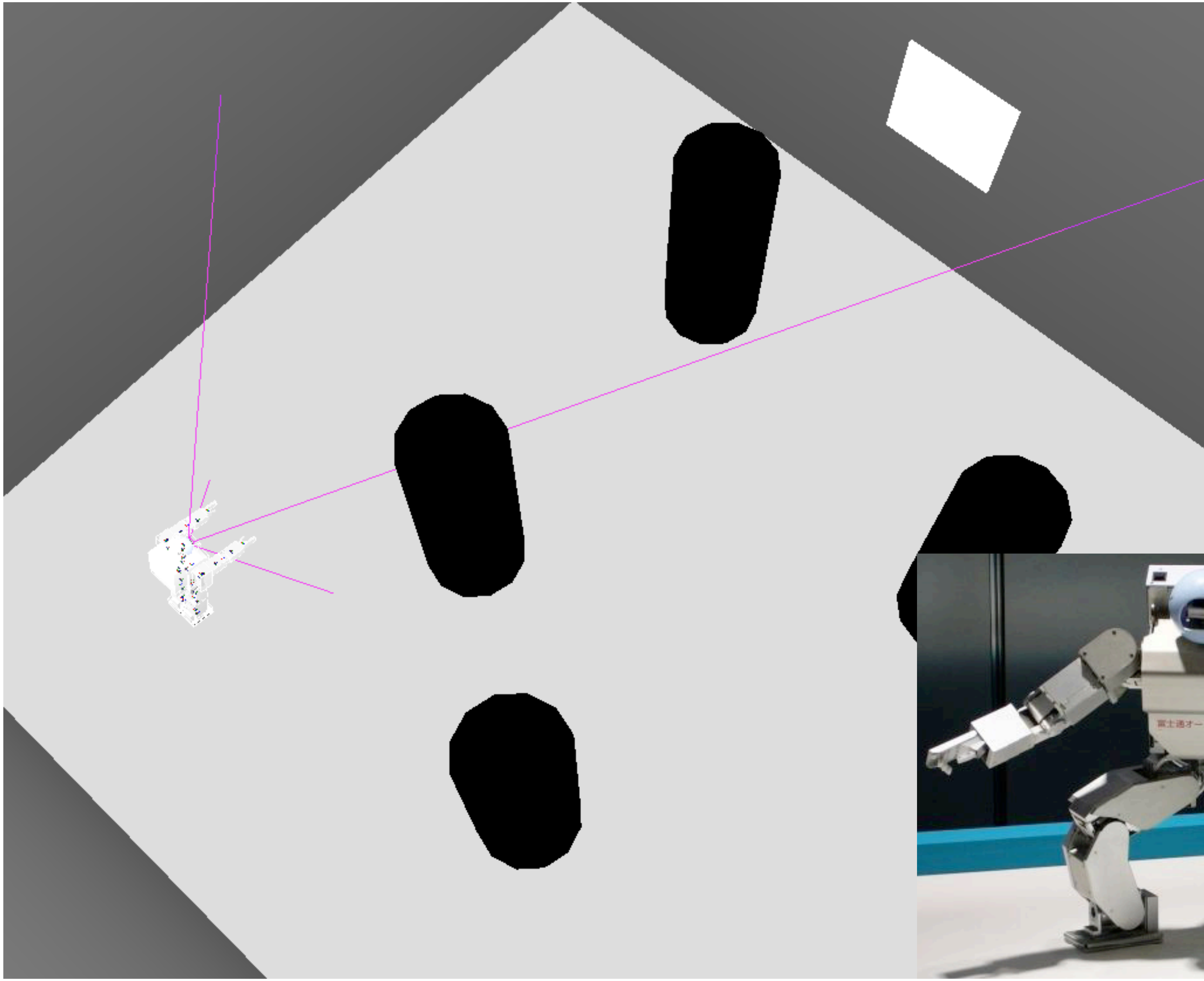
$$F = \frac{1}{R * M} \sum_{r=1}^{R} \sum_{m=1}^{M} D_{r,m}$$

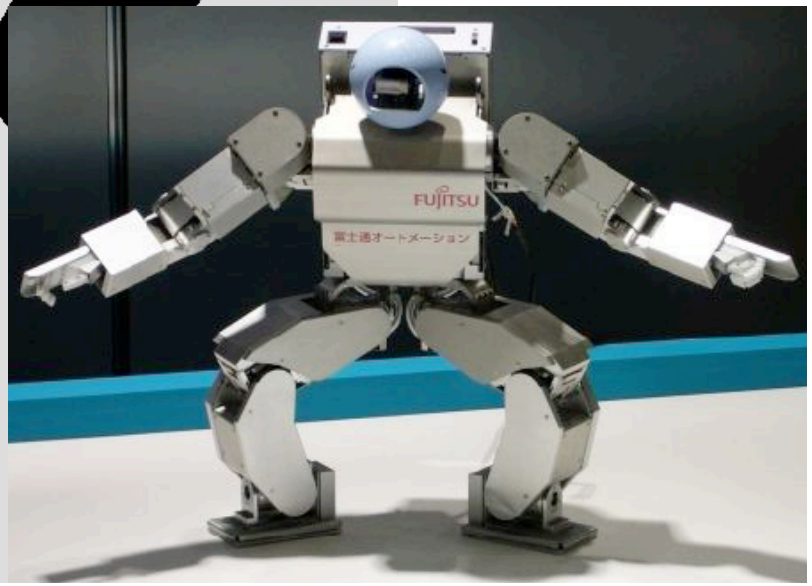




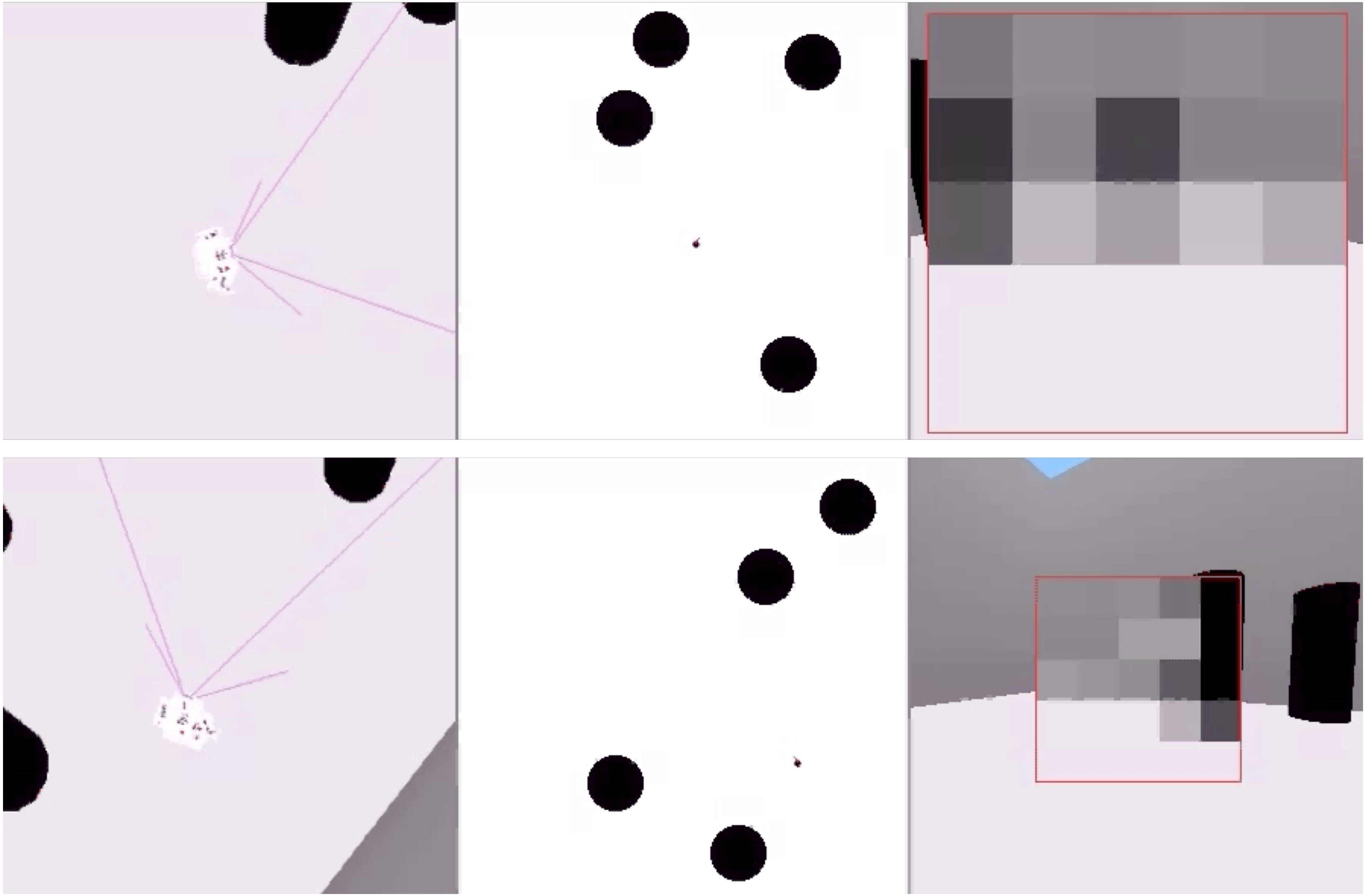

20.0fps

Vision based locomotion

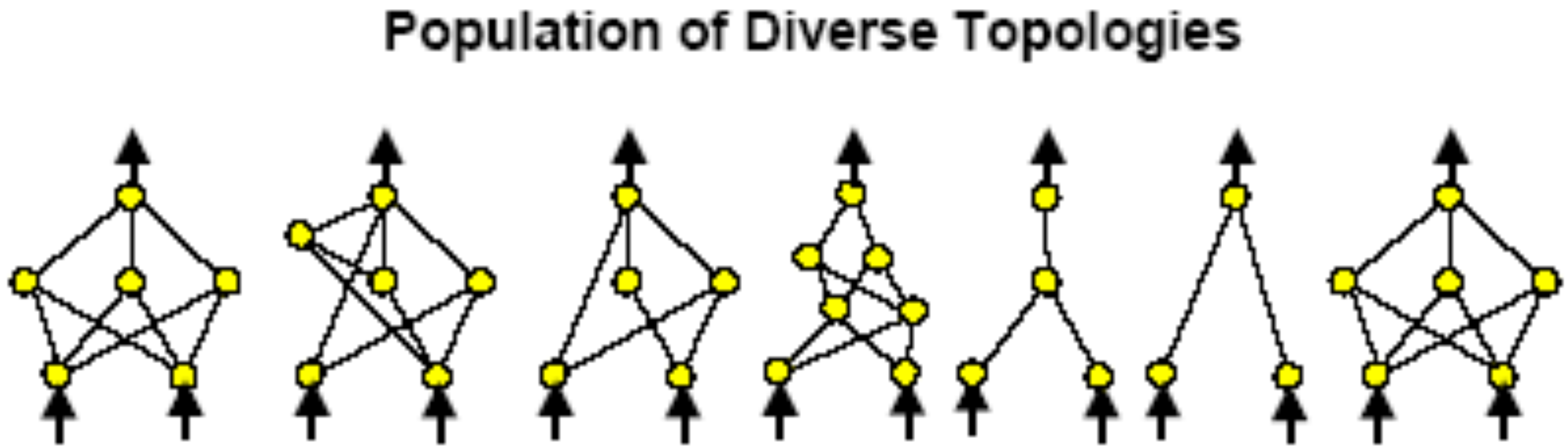




Vision based locomotion



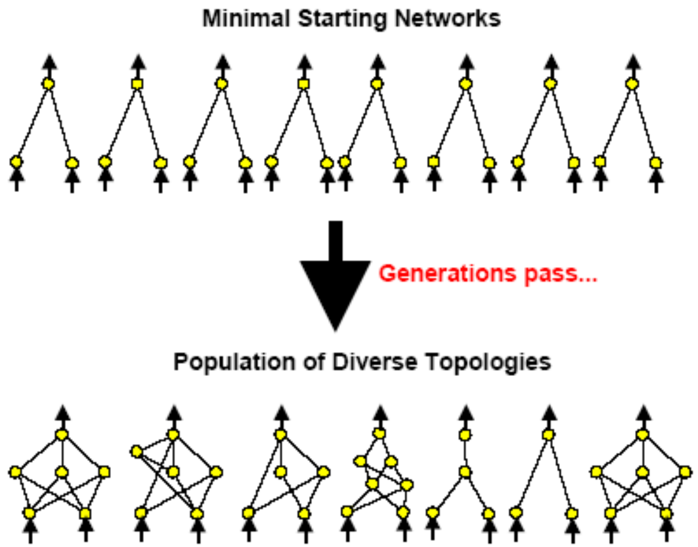
Evolution of Neural Topologies



Problems:

- Many non-functional network in initial population
- More complex networks can’t compete in the short run
- Competing conventions in genetic string

NeuroEvolution through Augmenting Topologies (NEAT)



PROBLEM: Compared to smaller networks, larger networks need more evolutionary time to obtain similar performance and could be eliminated from the population

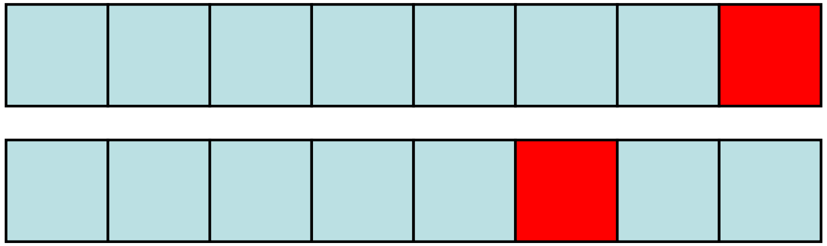
SPECIATION: continuously compare networks and group them by similarity, i.e. create species. Mating and crossover takes place only within the groups

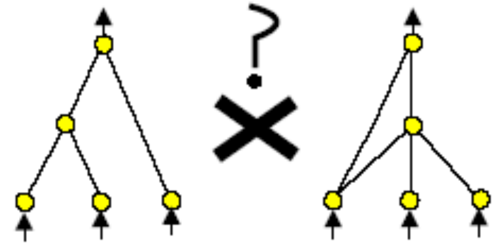
FITNESS SHARING: In order to prevent domination of a single group, individuals in a group share fitness: the fitness of the individual is divided by the number of individuals in the group (the larger the group, the smaller the individual fitness)

Stanley, Kenneth O., and Risto Miikkulainen. "Evolving neural networks through augmenting topologies." *Evolutionary computation* 10.2 (2002): 99-127.

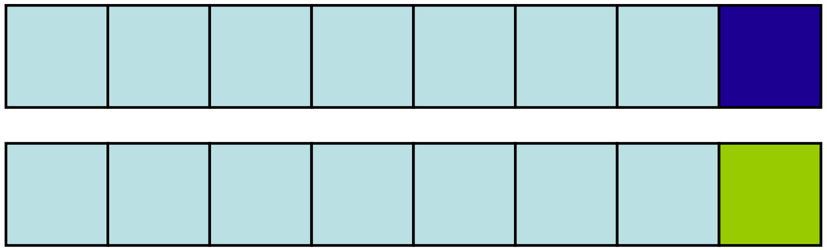
Competing conventions

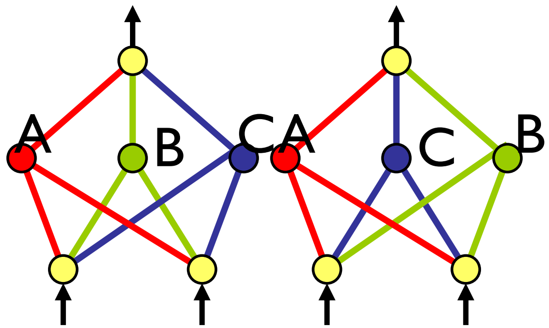
Same gene may be in different positions





Different genes may be in same positions

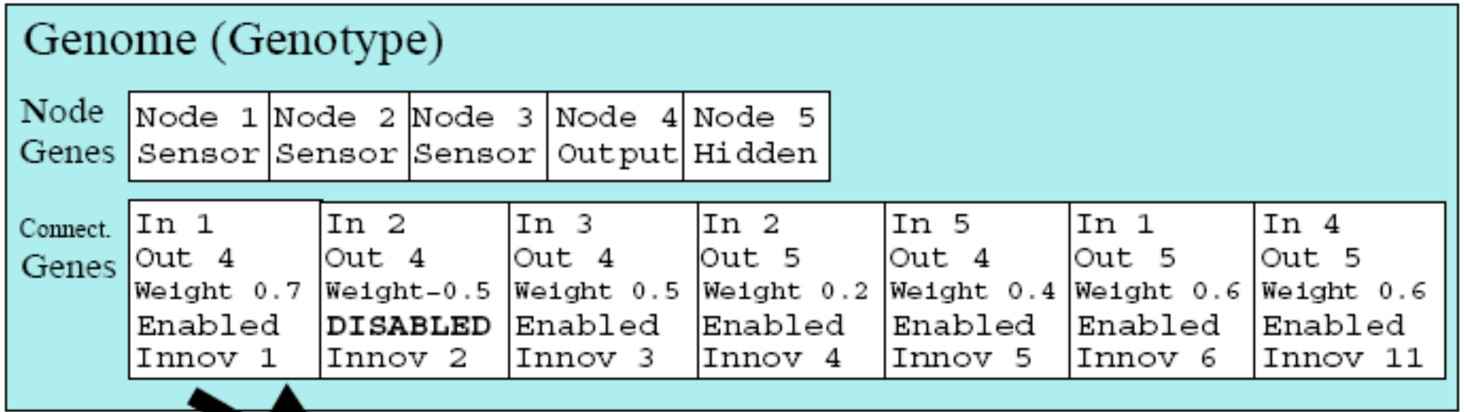


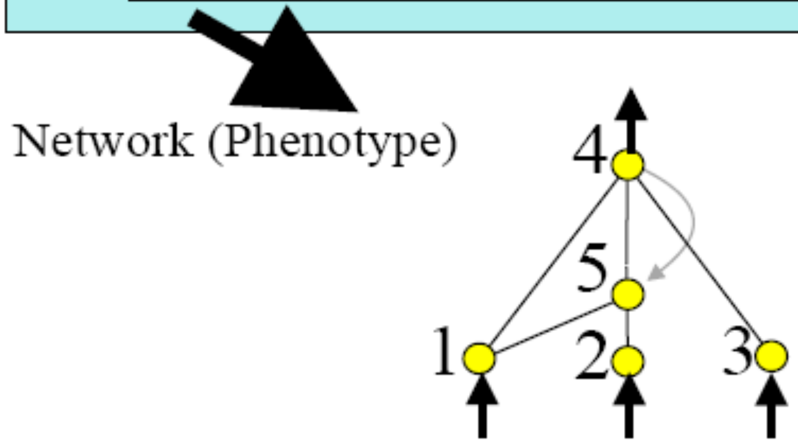


Stanley, Kenneth O., and Risto Miikkulainen. "Evolving neural networks through augmenting topologies." *Evolutionary computation* 10.2 (2002): 99-127.

NeuroEvolution through Augmenting Topologies (NEAT)

Use a global clock to keep track of “innovation time” in mutation

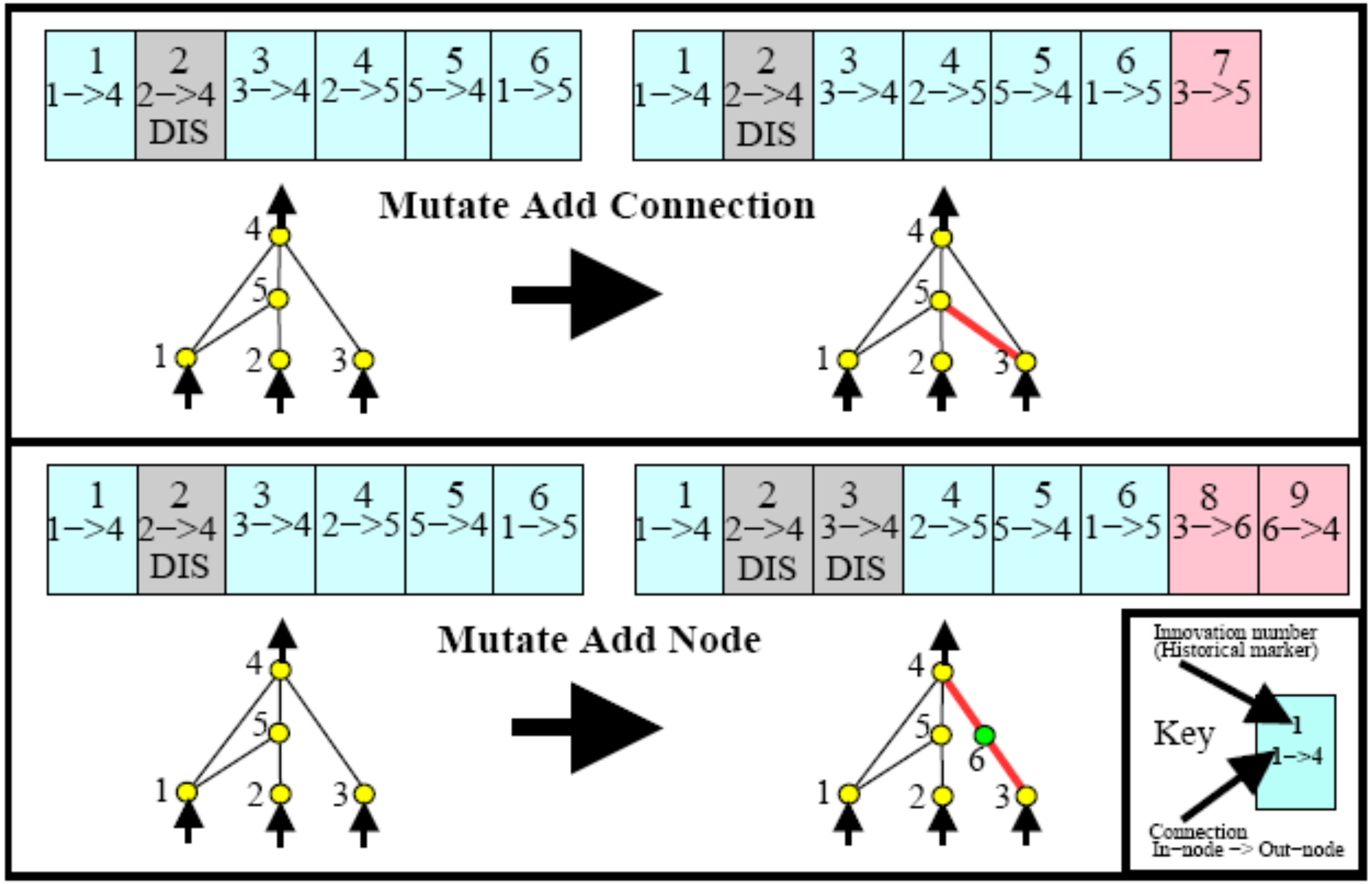




Innovation numbers tell exactly when in history a specific topological feature appeared, so that it can be matched up with the same feature in another network.

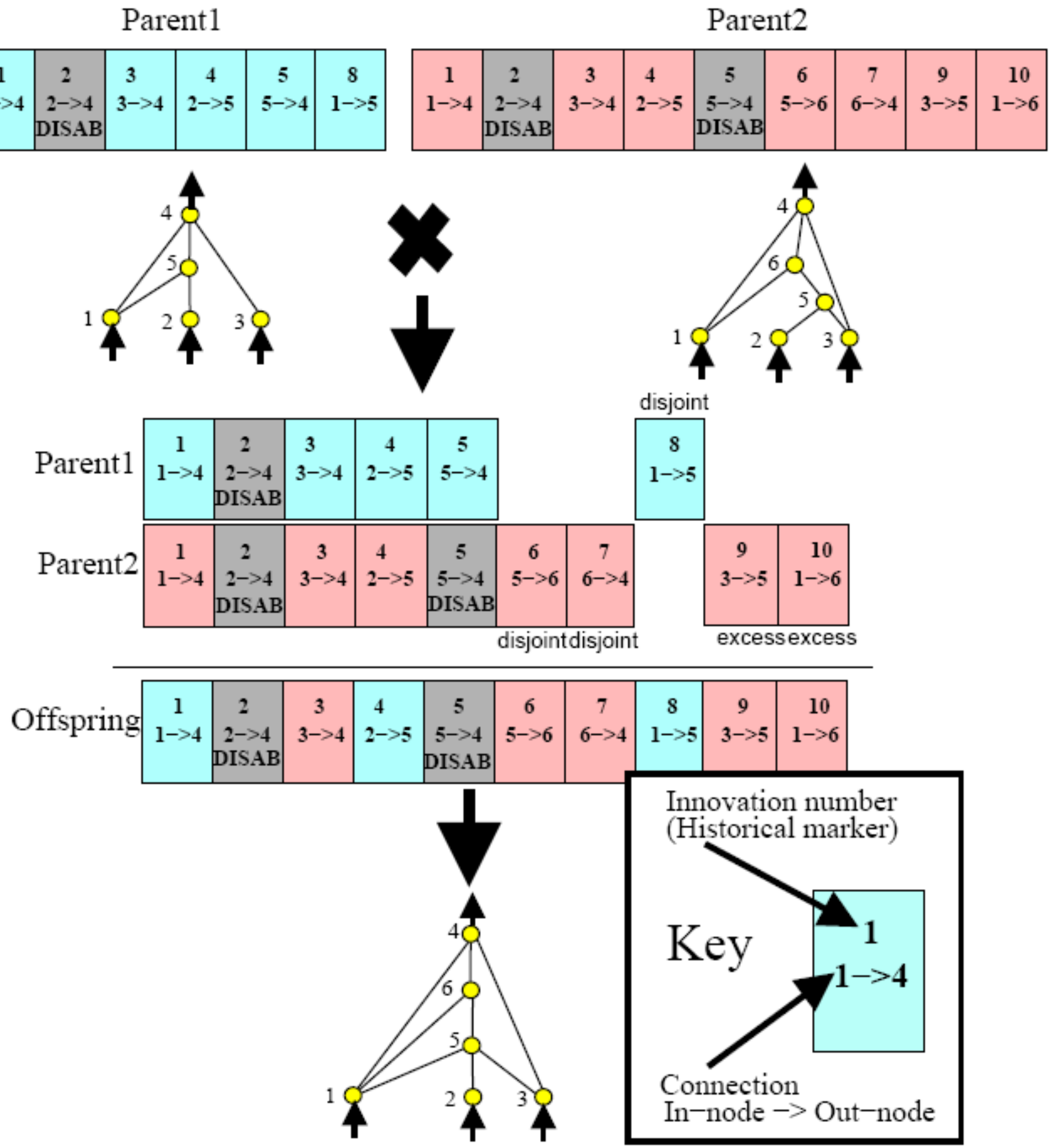
Stanley, Kenneth O., and Risto Miikkulainen. "Evolving neural networks through augmenting topologies." *Evolutionary computation* 10.2 (2002): 99-127.

Example of 2 innovations



Stanley, Kenneth O., and Risto Miikkulainen. "Evolving neural networks through augmenting topologies." *Evolutionary computation* 10.2 (2002): 99-127.

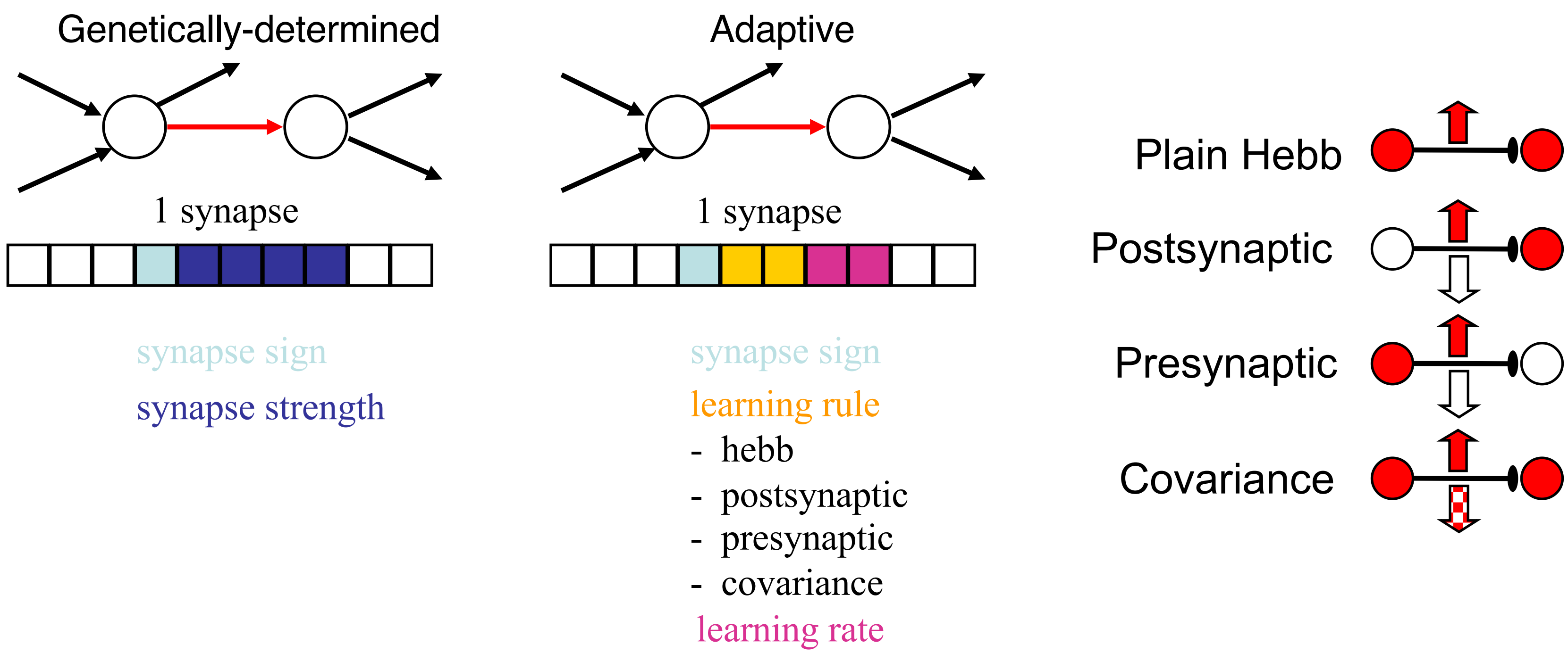
Crossover



Parent1
Parent2
1 1->4
2 2->4 DISAB
3 3->4
4 2->5
5 5->4
8 1->5
1 1->4
2 2->4 DISAB
3 3->4
4 2->5
5 5->4 DISAB
6 5->6
7 6->4
9 3->5
10 1->6
disjoint
disjoint disjoint
excess excess
Offspring
Key
Innovation number (Historical marker)
Connection In−node −> Out−node

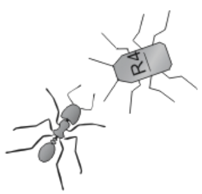
Evolution OF Learning

Synaptic learning rules can be genetically encoded. The rules are applied to the synaptic weights starting from random initial conditions.

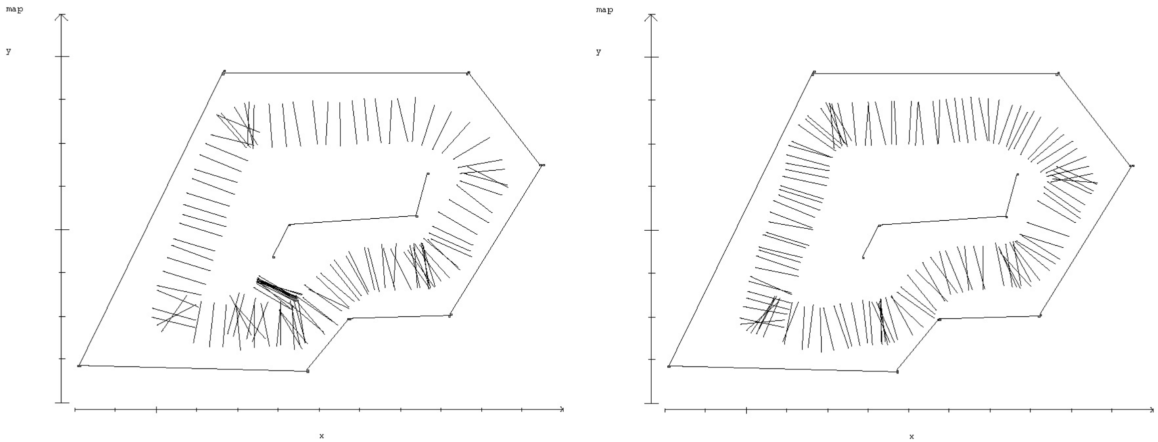


Important aspects:

- A neural network can use different learning rules in different parts
- There is no need of teacher or reinforcement learning, no gradient descent and local minima
- Individuals are selected for their ability to learn, not simply to solve a specific problem

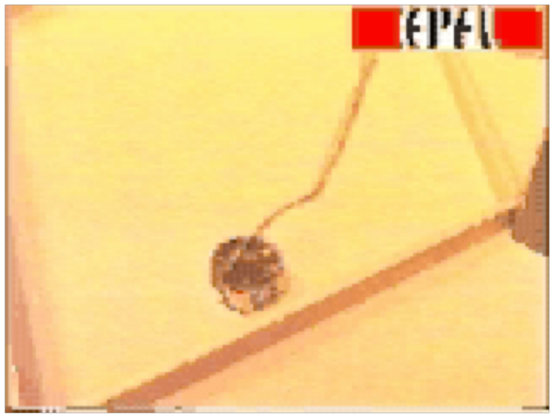


Online Adaptation

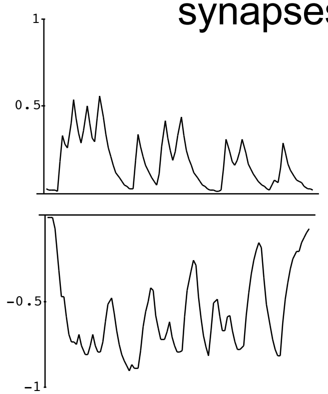


In addition, they perform well in different environments by developing suitable strategies. Contrary to conventional models, several synapses continue to change, but the overal pattern of change is dynamically stable.

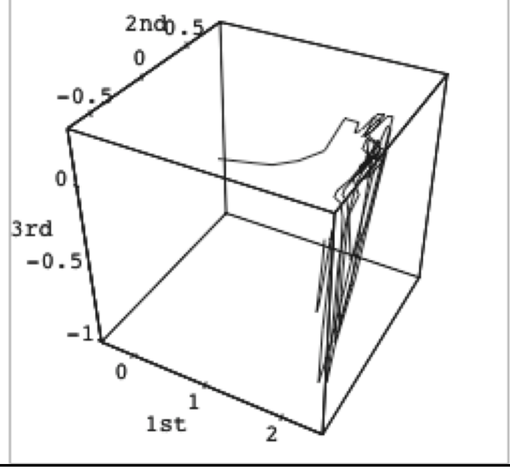
Test in new environment



Continuously changing synapses



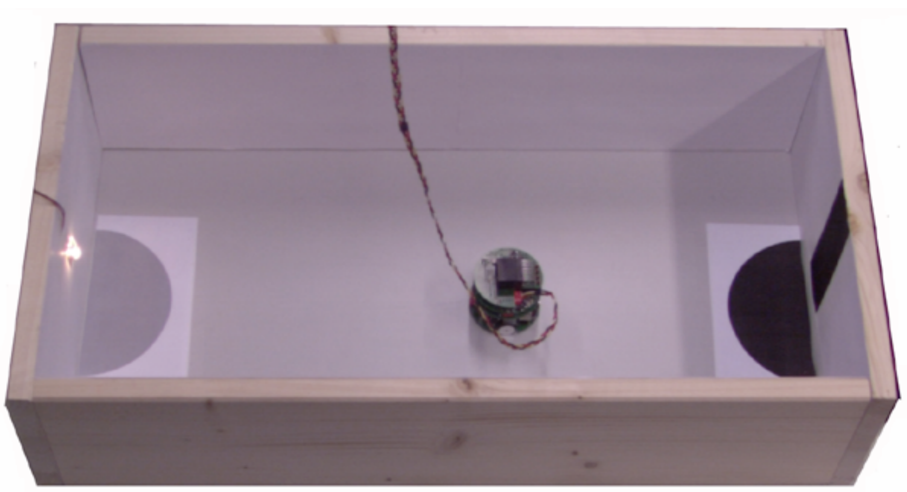
Dynamic stability

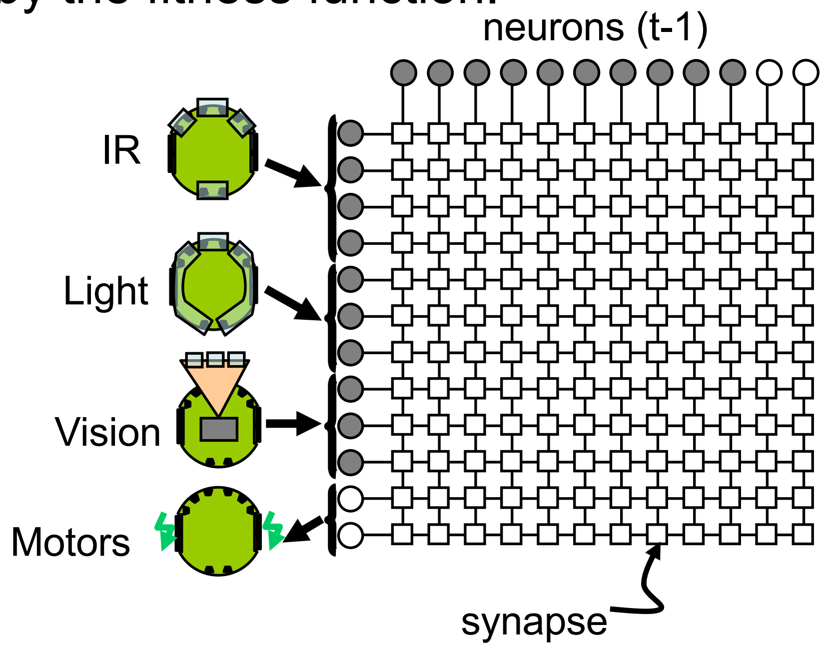


A Sequential Task

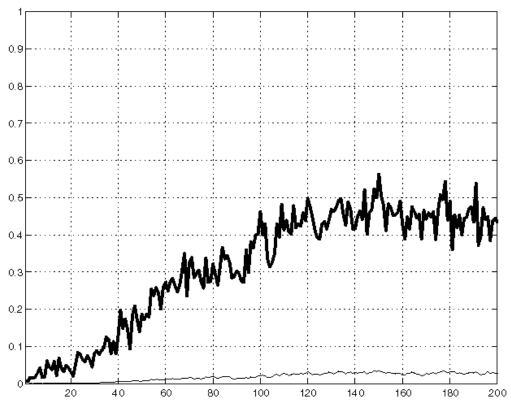
A Khepera robot is evolved to switch on a light and go under the light, but this sequence of actions is not directly rewarded by the fitness function.

Fitness= time_gray_light / total_time

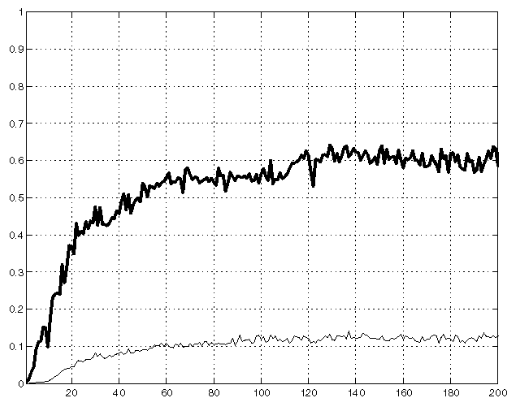




evolution of weights

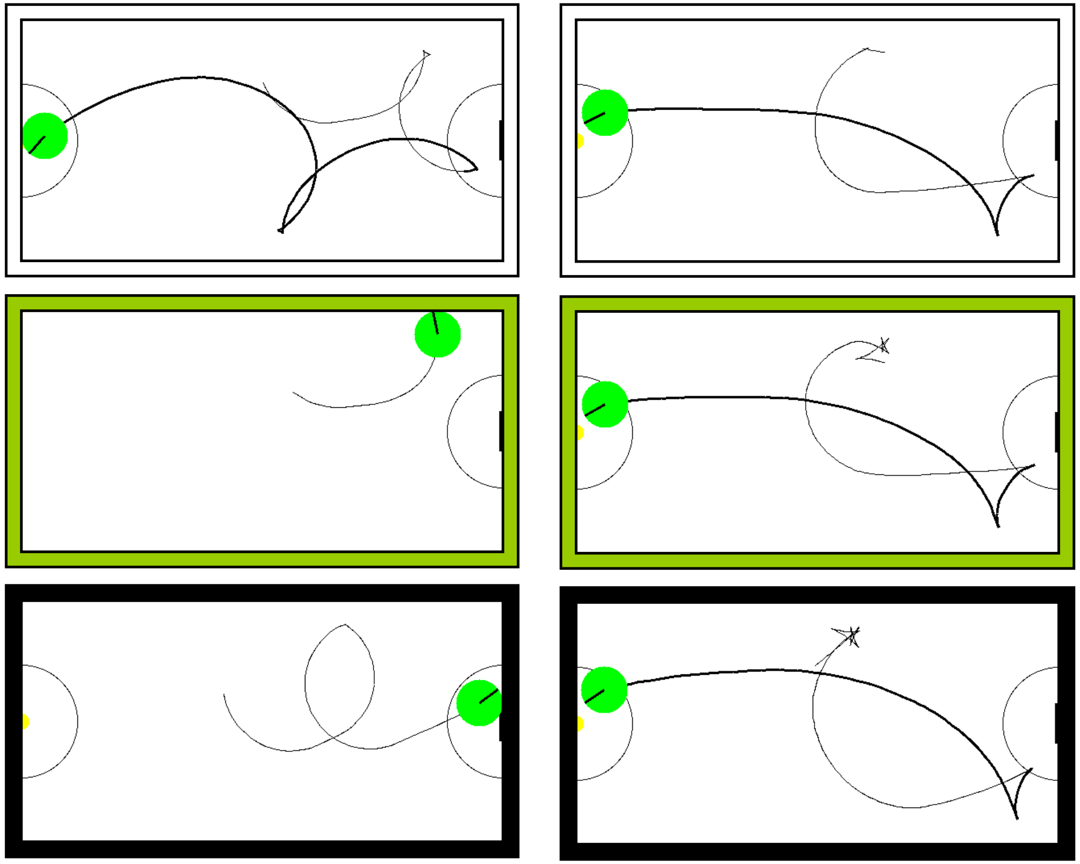


evolution of rules

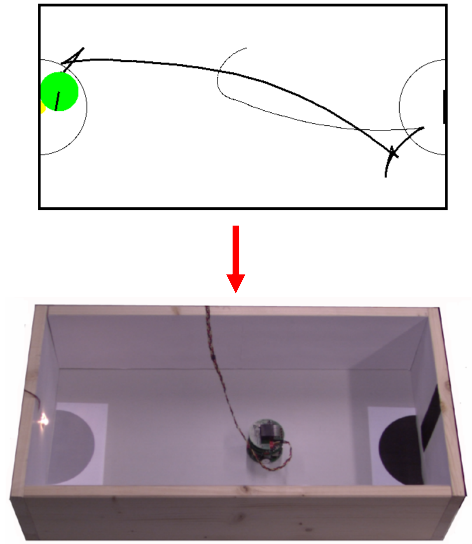


Robustness to Color Change

Evolved adaptive individuals can cope with new colours of the walls whereas genetically-determined individuals fail.



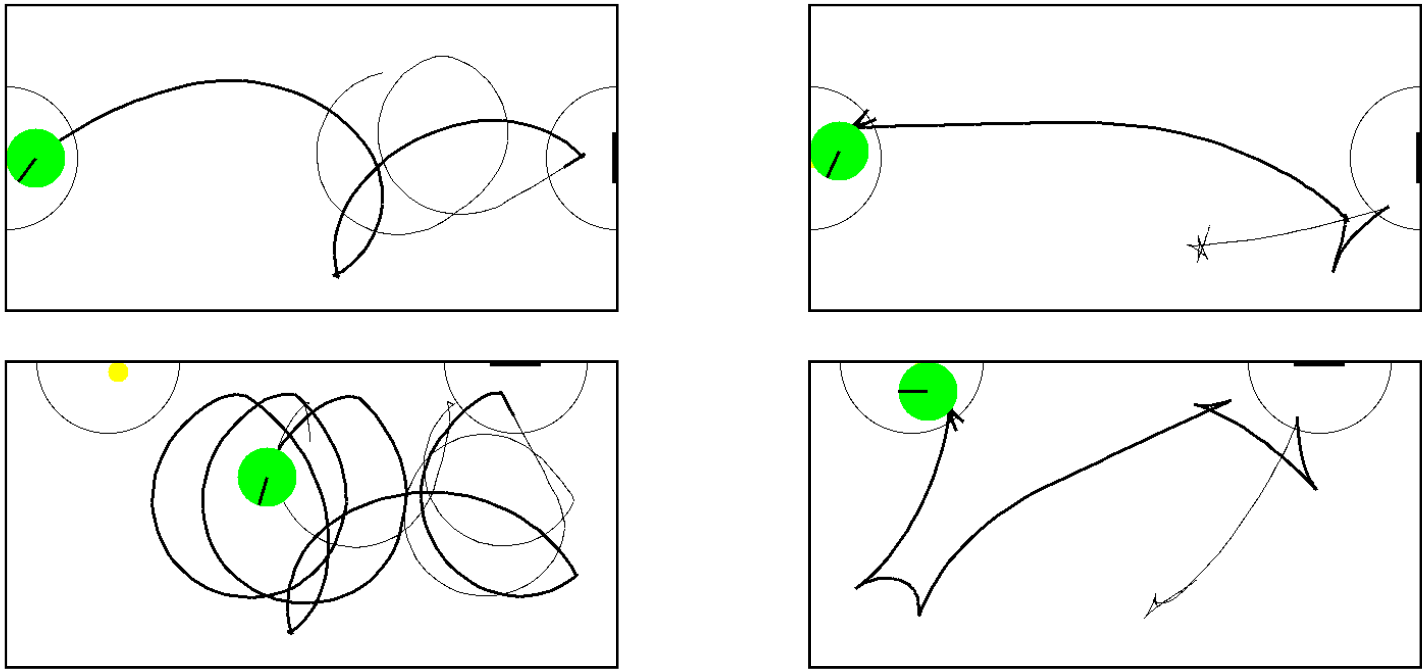
Similarly, evolved adaptive individuals transfer smoothly from simulated to real world.



Genetically-determined Adaptive

Robustness to Layout Modification

Evolved adaptive individuals can cope with new positions of the two landmarks whereas genetically-determined inviduals cannot.



Genetically-determined Adaptive

Fitness function design

Design of fitness functions that can generate *desired behaviors* is the most difficult aspect of Evolutionary Robotics.