

# Simultaneous Evolution of Morphology and Locomotion of Soft Robots by Novelty Search

Georgios Methenitis

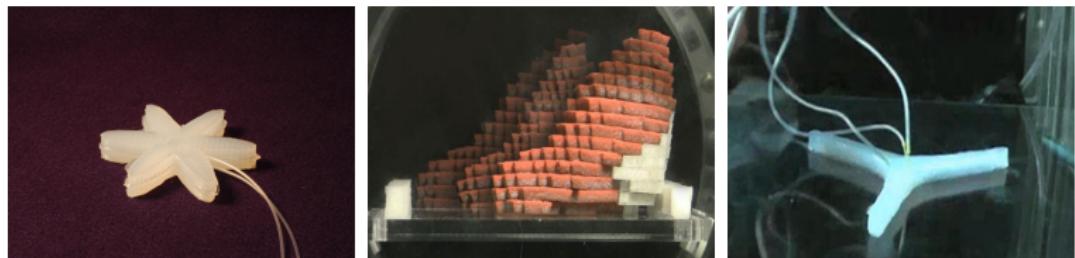
UvA, ACT

July 21, 2014

# Introduction

## Soft Robots

- ▶ Inspired by nature
- ▶ Completely soft bodies
- ▶ Capable of developing new kinds of locomotion



Soft robots can be actuated through air pressure tubes, environmental changes ( temperature, pressure ), even explosions.

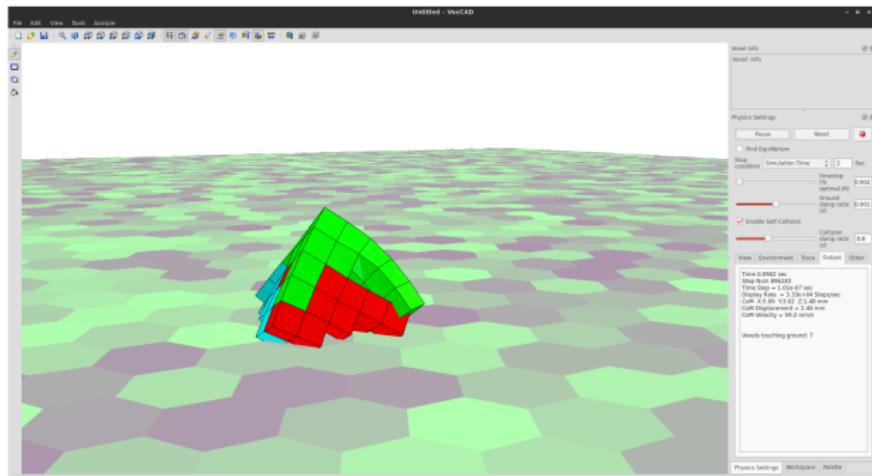
# Related Material



## VoxCad Simulator [2]

- ▶ Created by Jonathan Hiller and Hod Lipson
- ▶ Voxel modeling and analyzing software

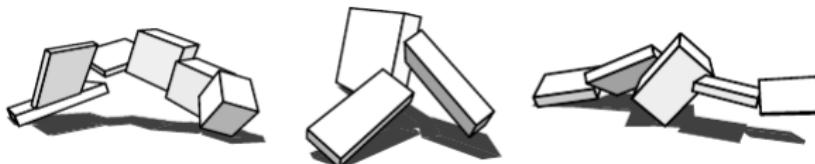
- ▶ Lattice
- ▶ Voxels
- ▶ Structure
- ▶ Materials



## Related Work |

### *Evolving virtual creatures [5]*

- ▶ Rigid body parts, joints
- ▶ Evolution of the morphology and the control



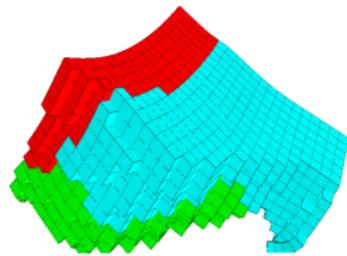
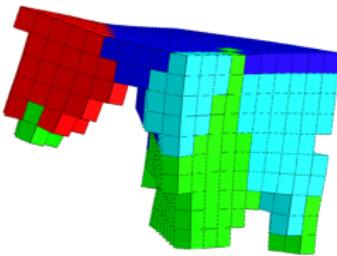
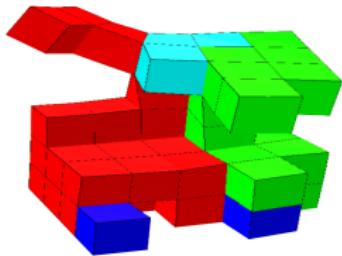
### *Evolving a diversity of virtual creatures through novelty search and local competition [4]*

- ▶ Same experimental framework
- ▶ Novelty < Fitness
- ▶ Novelty search with global competition has the best average fitness.

## Related Work II

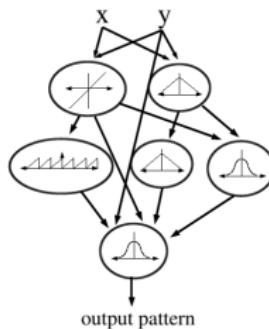
*Evolving soft robots with multiple materials and a powerful generative encoding.* [1]

- ▶ Generative encoding, Compositional pattern-producing network, CPPN.
- ▶ Neuroevolution of augmenting topologies, NEAT.



## Compositional pattern-producing network [7]

- ▶ Similar to artificial neural networks
- ▶ Different set of activation functions



- ▶ Produce symmetrical and repetitive patterns
- ▶ Appropriate for problems with geometrical structure

## NeuroEvolution through Augmented Topologies (NEAT) [6]

Some key points of this method are:

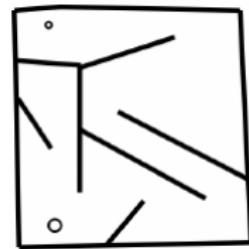
- ▶ Evolving neural network topologies along with weights
- ▶ Crossover between different topologies
- ▶ Structural innovation through speciation (New species have time to improve)

# Novelty Search

What is novelty search:

- ▶ Traditionally fitness measures how good an individual is (Objective function).
- ▶ Objective function can prevent evolution reaching the global maximum.
- ▶ Abandon the objective.
- ▶ Try finding novelty in behavior space.
- ▶ Random?

Definition (Sparsity)



$$s(x) = \frac{1}{k} \sum_{i=0}^k dist(x, b_i)$$

# Research Topics

- ▶ Gravity
  - ▶ Performance under different conditions of gravity
- ▶ Novelty search
  - ▶ Performance, in respect to the original fitness
  - ▶ Performance, in behavior space
  - ▶ Behavior, what is a good behavior metric?
- ▶ Other evolutionary algorithms
  - ▶ Genetic algorithm with direct coding
  - ▶ Random generative encoding
  - ▶ Covariance Matrix Adaptation Evolution Strategy (CMA-ES)
  - ▶ Differential Evolution (jDE)
- ▶ Can we evolve CPPNs with other evolutionary algorithms?

## Things completed so far...

- ▶ Replication of the results from [1]
- ▶ Generative random encoding
- ▶ Simple genetic algorithm
- ▶ Implementation of CPPN-NEAT experiment  
(HyperNEAT C++ library)
- ▶ Novelty search
- ▶ Competition between species (novelty, fitness)
- ▶ Combination of novelty and fitness

# Random Soft Robots

## Random

Assign materials to voxels randomly.

## Generative encoding

Only two parameters can change in this encoding.

1. The probability of adding a new voxel into the structure.
2. The probability that the new voxel introduced will use the same kind of material as its connection.

## Random CPPNs

Evolution with high mutation power and no fitness information.

## Simple genetic algorithm

- ▶ GAlib C++ library
- ▶ Each genome is represented by a stream of real numbers in  $[0, 1]$ .
- ▶ The length of this stream is equal to:

$$l = n \times (m + 1)$$

, where  $n$ , is the number of total voxels and  $m$  is the number of materials.

- ▶ For a lattice's dimensions of  $10 \times 10 \times 10$  and 4 materials the length of the genome is 5000.
- ▶ Simple genetic algorithm fails to produce locomotion.
- ▶ No structure knowledge.

## CPPN-NEAT

- ▶ HyperNEAT C++
- ▶ Each genome is represented by a CPPN.
- ▶ This CPPN is queried for each input coordinate to output the existence and the type of the material.
- ▶ NEAT evolves these CPPNs.
- ▶ In each generation, speciation. Population is split into species, new species can survive easier than old.

## CPPN-NEAT with Novelty Search [3]

- ▶ Same code base
- ▶ Novelty takes the place of fitness
- ▶ Novel individuals stored in a list
- ▶ For each new individual in the population, check its novelty in respect to the stored novel individuals.
  - ▶ Sparsity

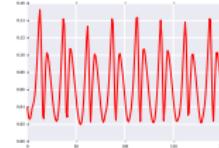
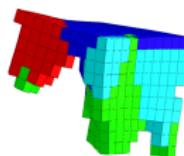
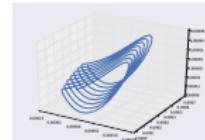
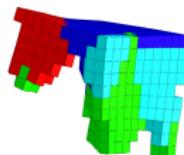
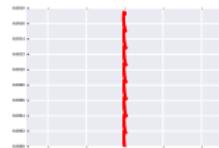
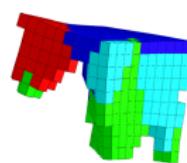
# Behavior

How can we go from fitness to behavior:



$$\text{input patterns} \rightarrow \text{3D volume} \rightarrow F(x) \rightarrow B(x)$$

Examples:



# Behavior

Behavior types can be used:

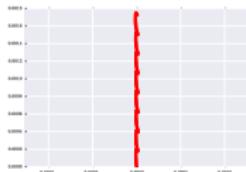
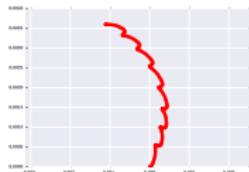
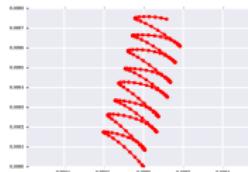
- ▶ Trajectory 3D, 2D
- ▶ Pace
- ▶ Voxels touching ground
- ▶ Kinetic energy
- ▶ Maximum pressure

Behavior similarity can be computed:

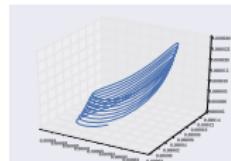
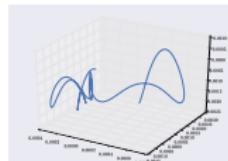
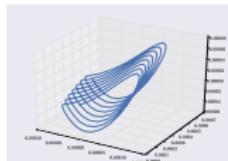
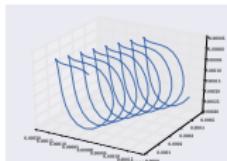
- ▶ Sum of Euclidean distances per timestep
- ▶ Cross-correlation

# Behavior Examples I

## 2D - Trajectories:

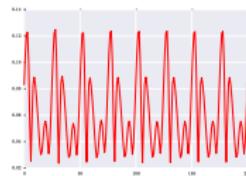
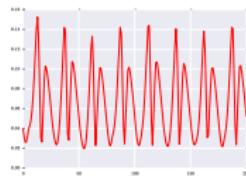
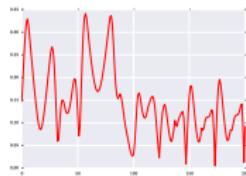
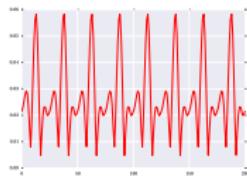


## 3D - Trajectories

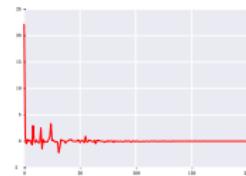
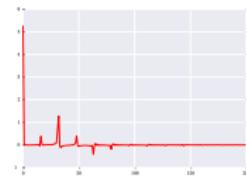
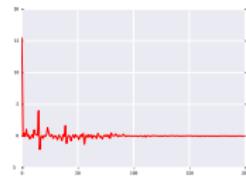
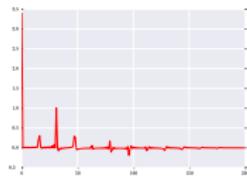


# Behavior Examples II

Pace per timestep

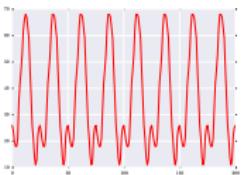
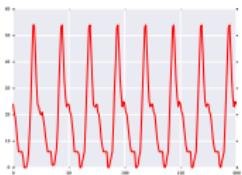
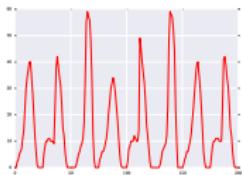
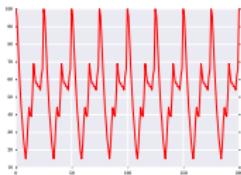


Pace - DFT

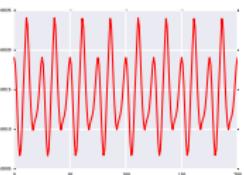
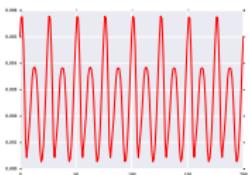
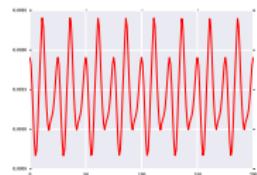
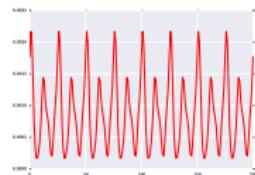


# Behavior Examples III

Voxels touching ground per timestep

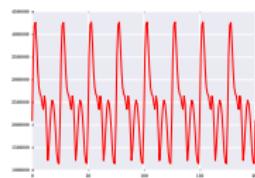
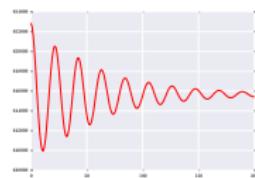
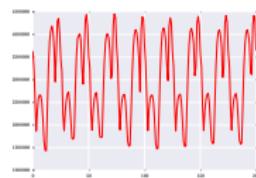
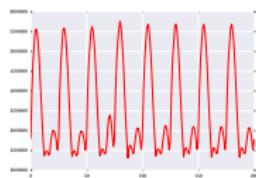


Kinetic energy per timestep



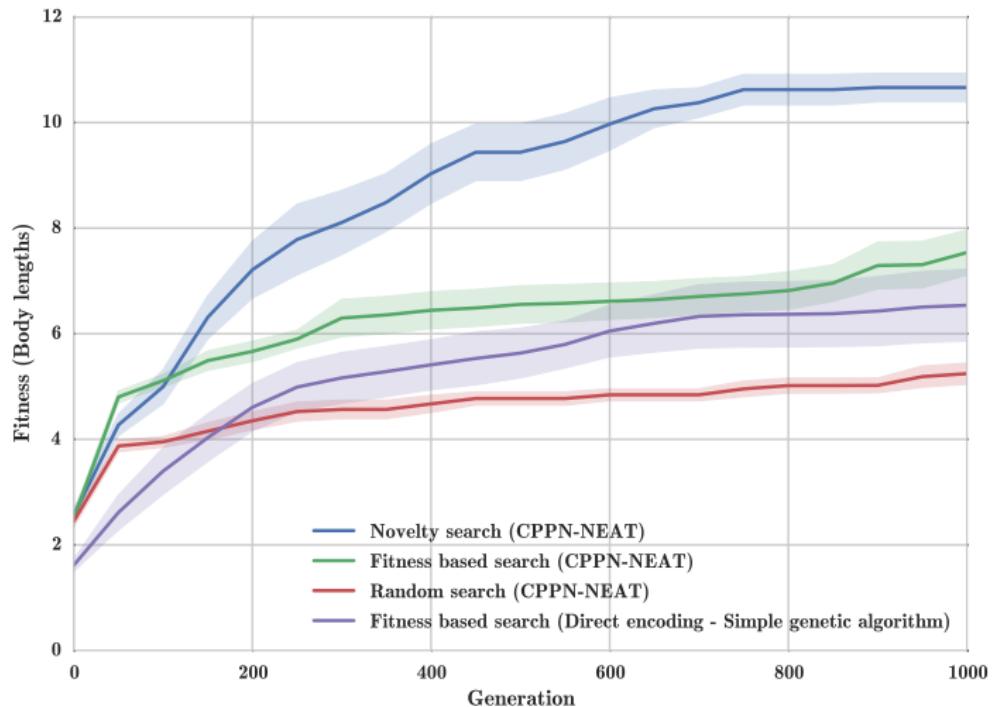
# Behavior Examples IV

Maximum pressure per timestep



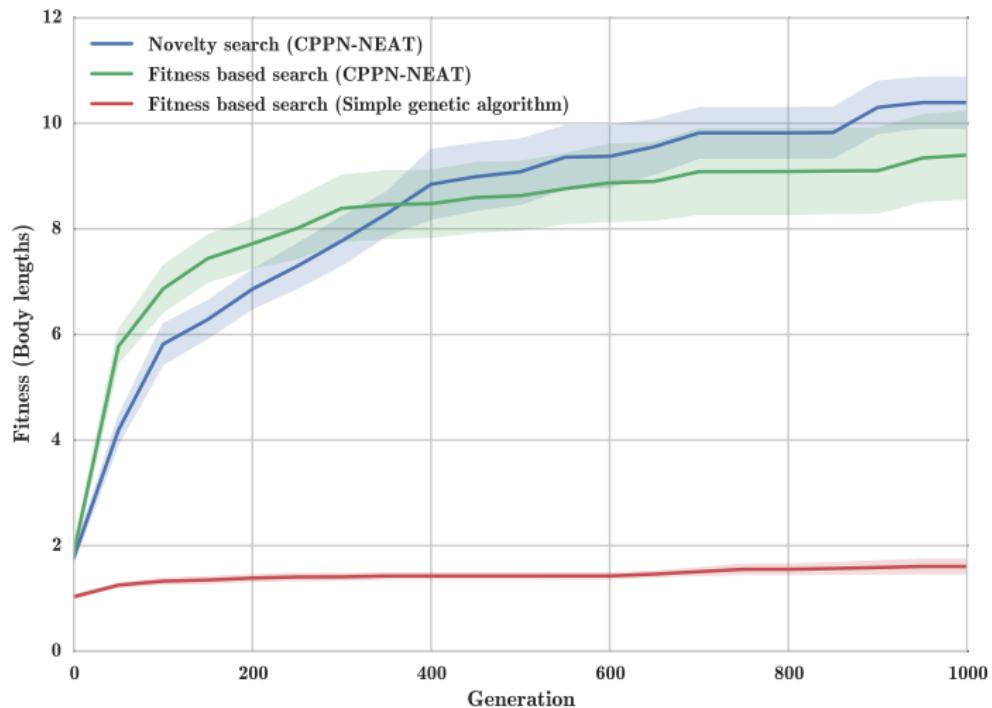
# Results I

## Fitness, Novelty, Random, Direct GA



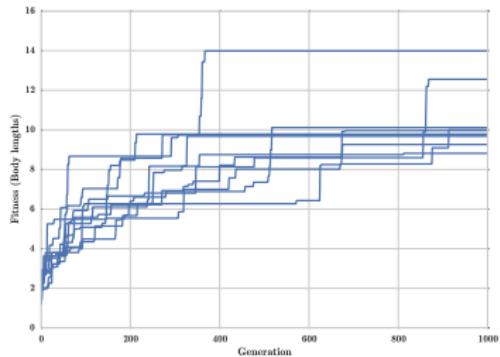
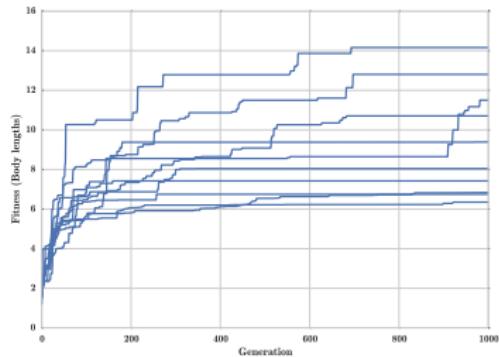
## Results II

### Fitness, Novelty, Direct GA



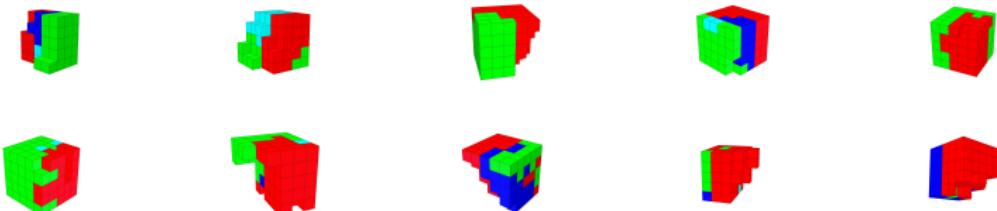
## Results III

10 individual runs for fitness-novelty based search.

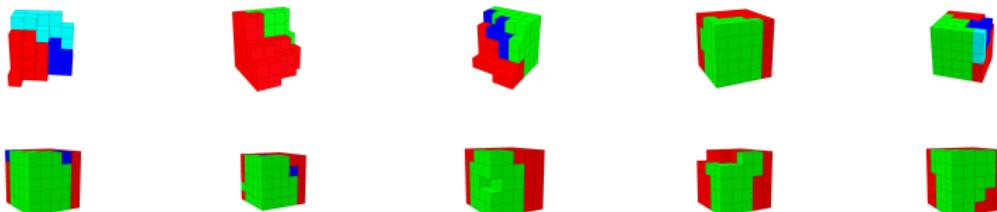


## Results IV

Novelty Search - Champions every 100 generations:

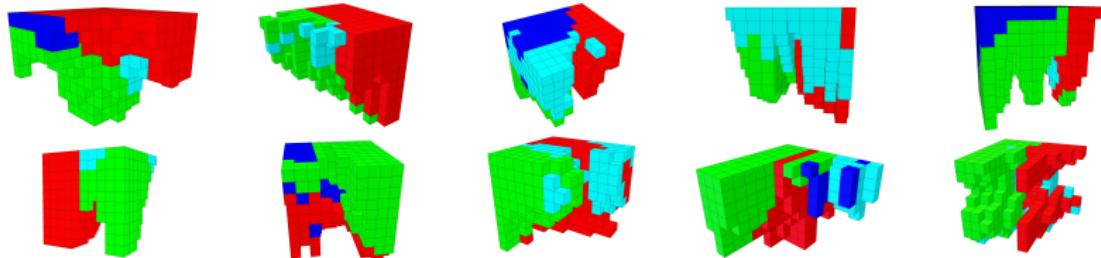


Fitness-based Search - Champions every 100 generations:

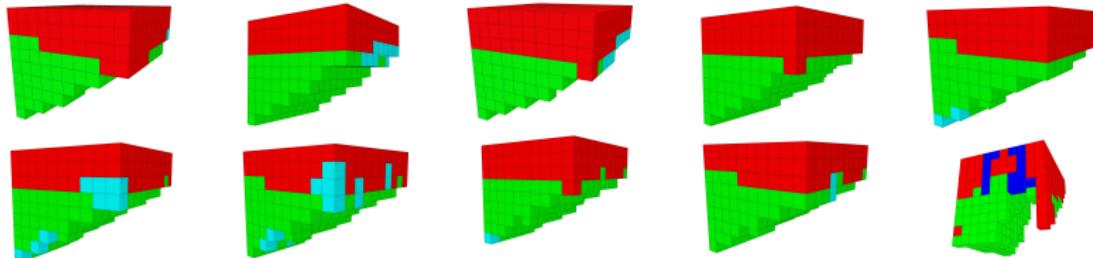


## Results V

Novelty Search - Champions every 100 generations:

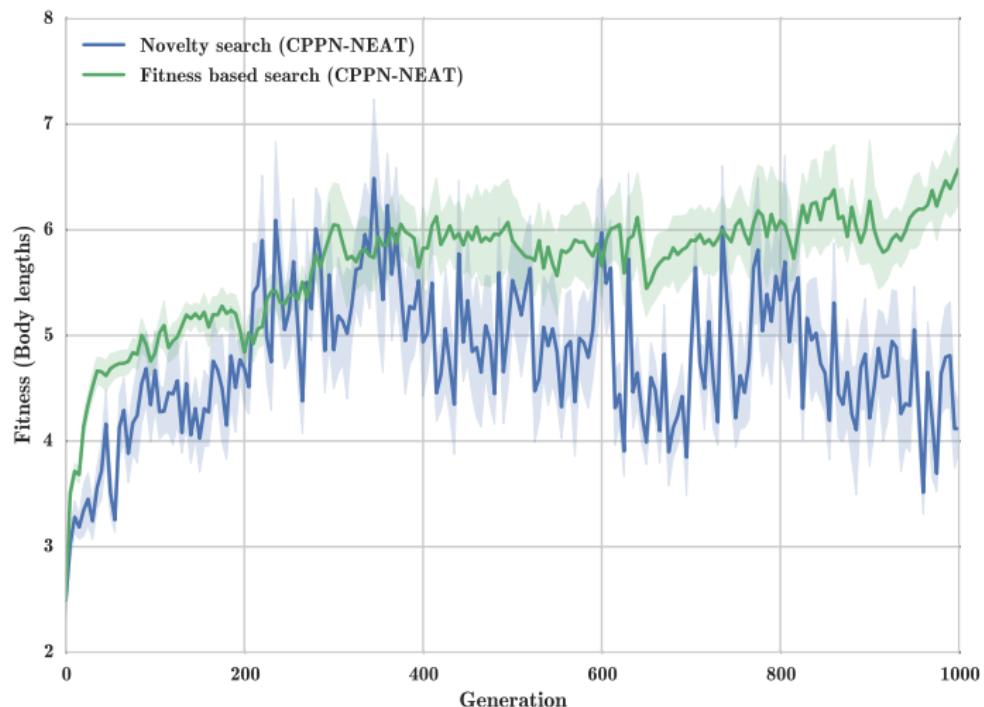


Fitness-based Search - Champions every 100 generations:



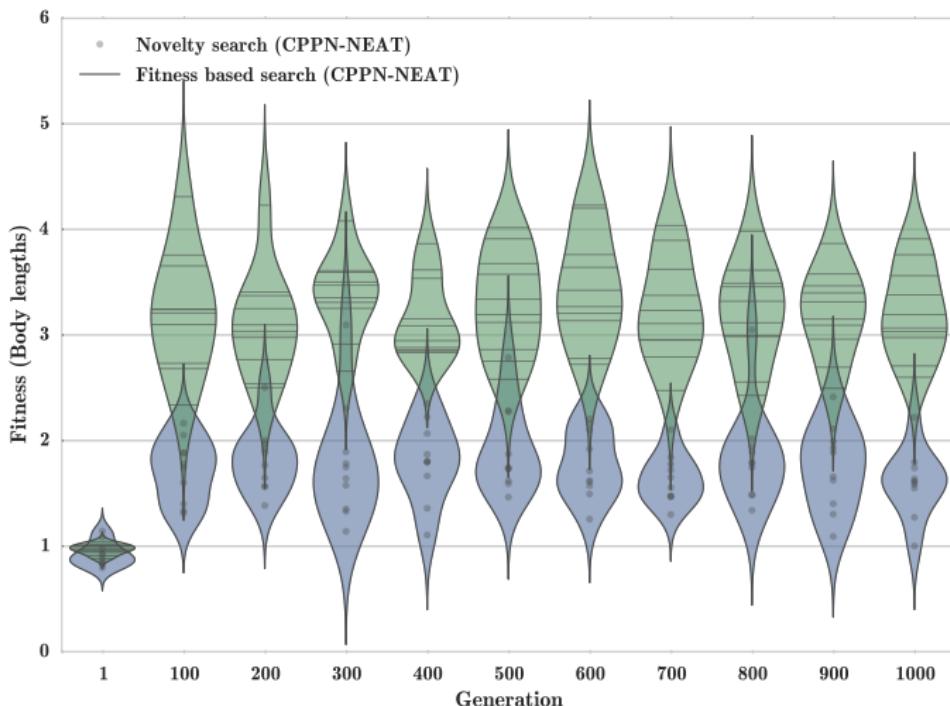
## Results VI

Fitness of the generation champion (best individual) per generation



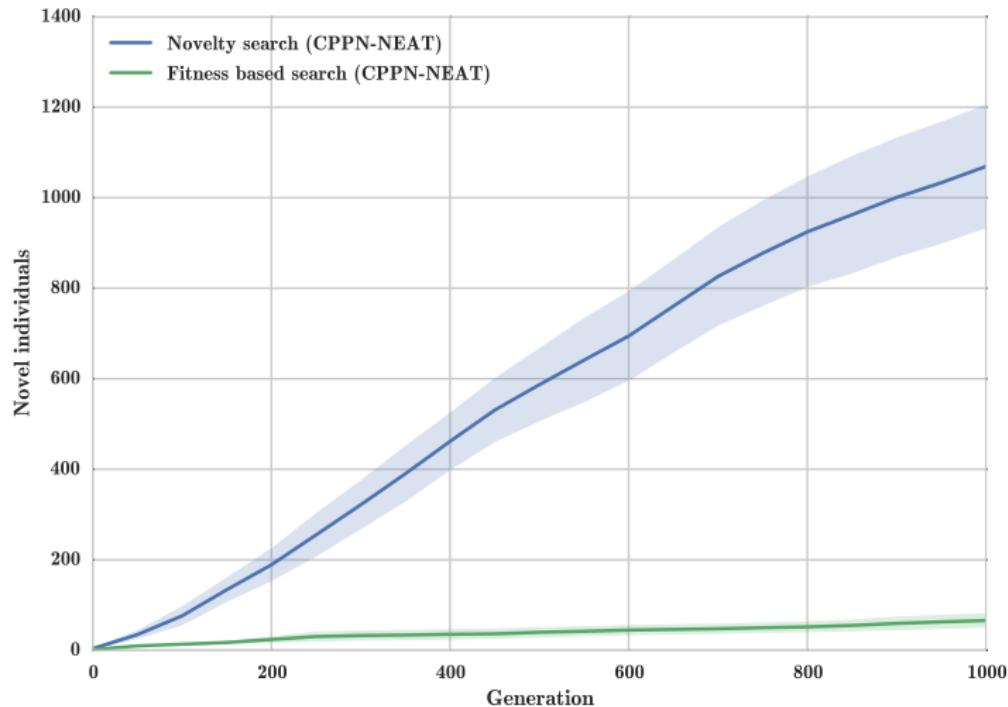
## Results VII

Novelty Vs. Fitness, distributions of average population's fitness per generation



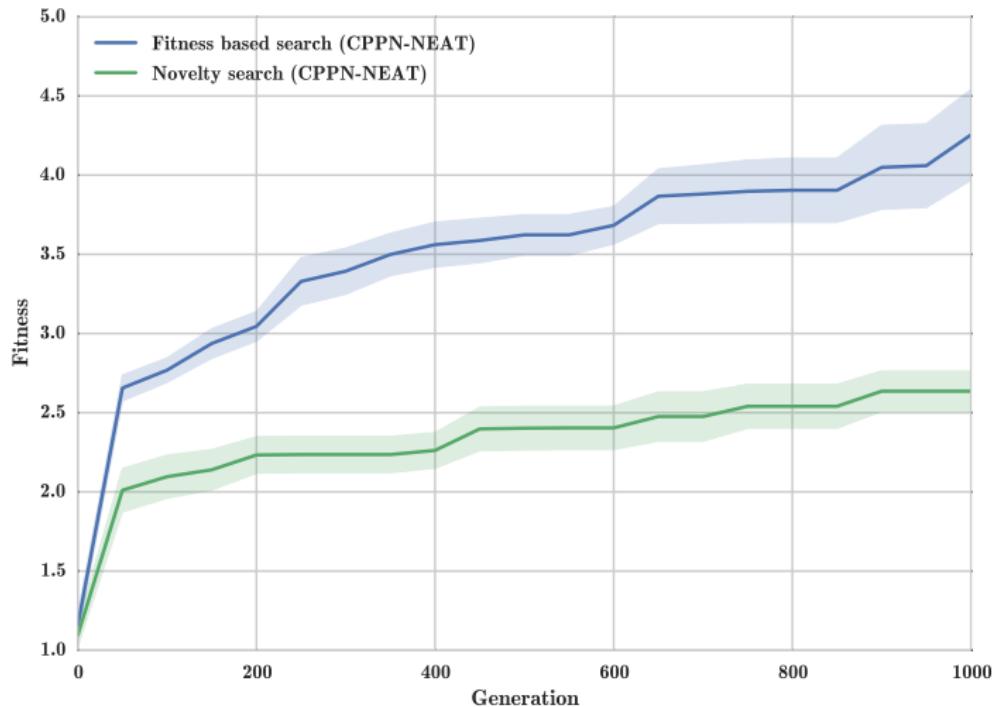
## Results VIII

### Novelty Vs. Fitness, number of novel behaviors



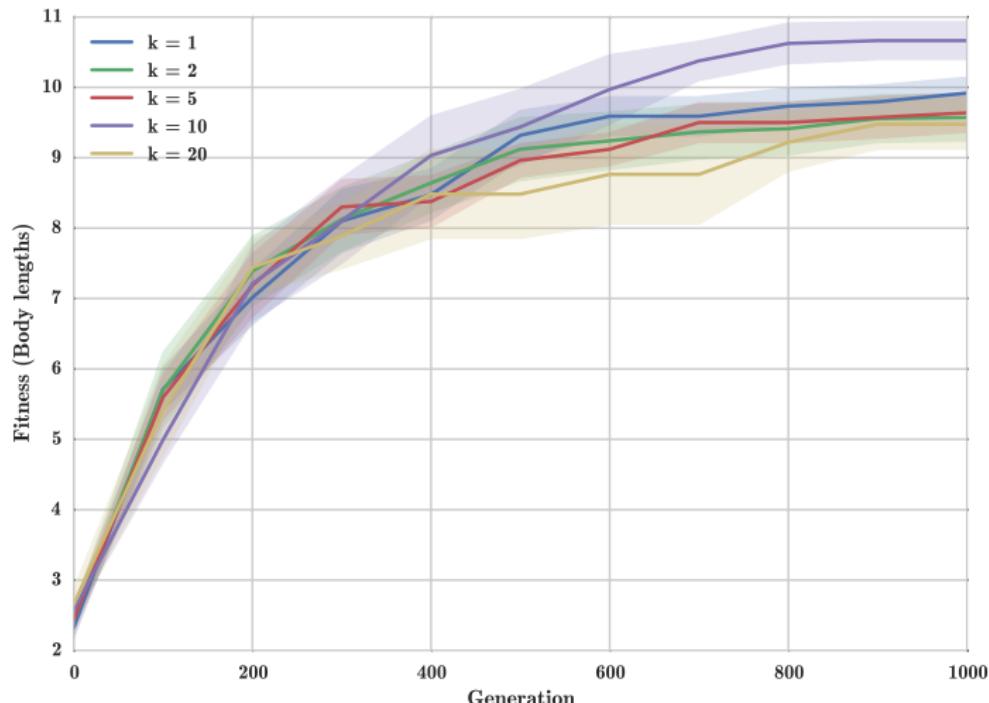
# Results IX

Penalizing actuated materials, *Behavior*: 2D trajectories



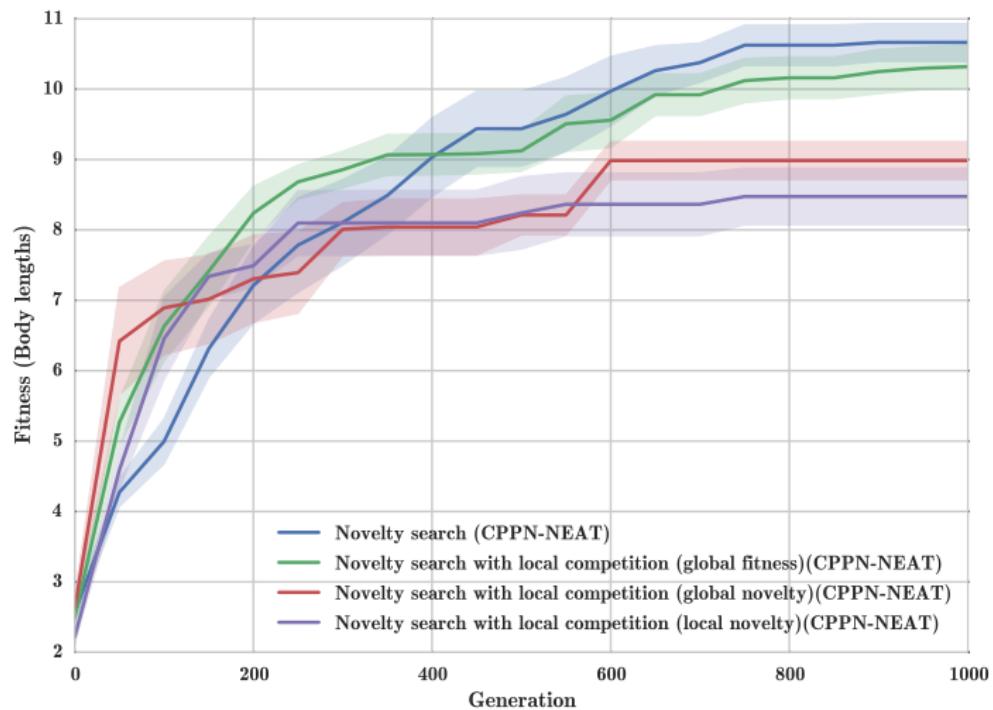
## Results X

Novelty search - The novelty is computed as the average distance from the  $K$ -nearest behaviors.



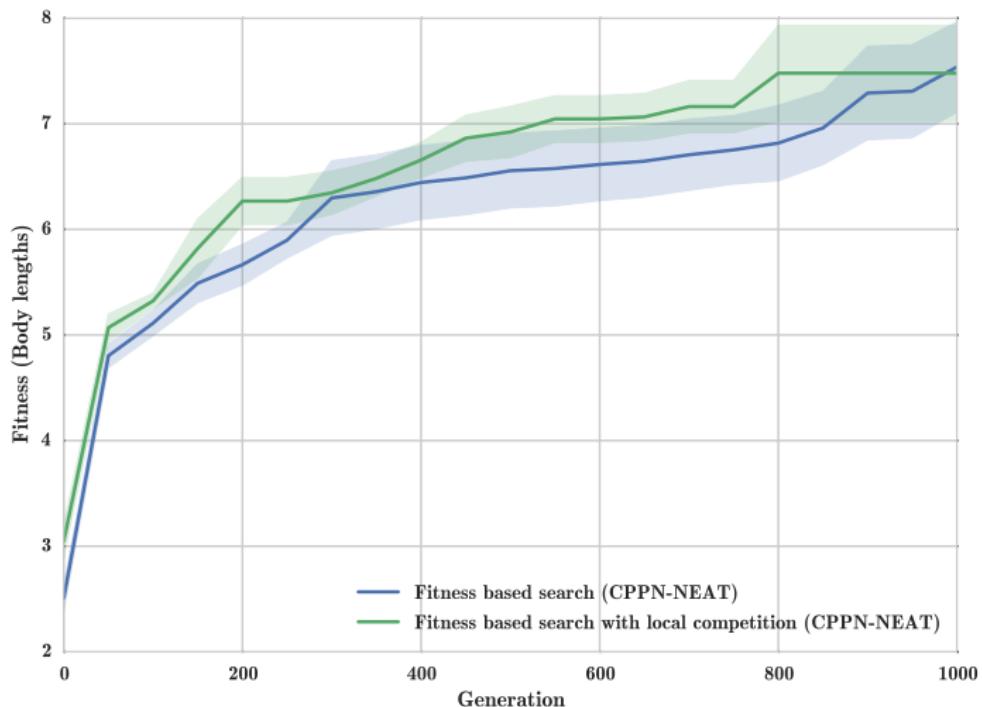
## Results XI

### Local competition (global fitness, global novelty, local novelty)

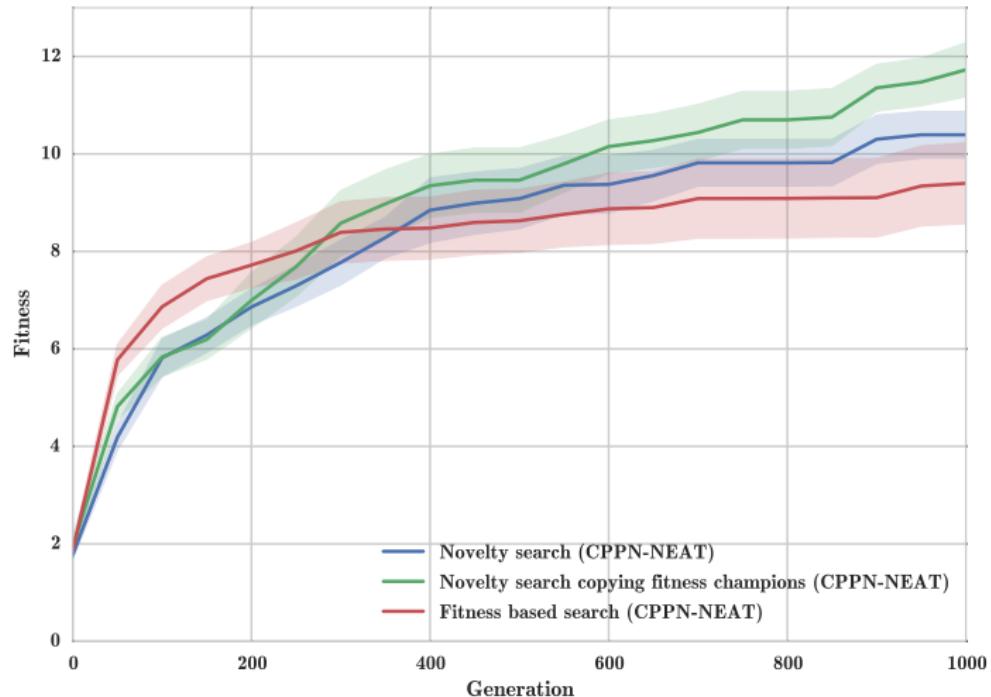


## Results XII

Local competition is held among the complete population of each species.



## Results XIII



## References I

-  Nick Cheney, Robert MacCurdy, Jeff Clune, and Hod Lipson.  
Unshackling evolution: evolving soft robots with multiple materials and a powerful generative encoding.  
In *Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference*, pages 167–174. ACM, 2013.
-  Jonathan Hiller and Hod Lipson.  
Dynamic simulation of soft heterogeneous objects.  
*arXiv preprint arXiv:1212.2845*, 2012.
-  Joel Lehman and Kenneth O Stanley.  
Abandoning objectives: Evolution through the search for novelty alone.  
*Evolutionary computation*, 19(2):189–223, 2011.

## References II

-  **Joel Lehman and Kenneth O Stanley.**  
Evolving a diversity of virtual creatures through novelty search and local competition.  
In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 211–218. ACM, 2011.
-  **Karl Sims.**  
Evolving virtual creatures.  
In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 15–22. ACM, 1994.
-  **Kenneth Stanley and Risto Miikkulainen.**  
Evolving neural networks through augmenting topologies.  
*Evolutionary computation*, 10(2):99–127, 2002.

## References III

-  Kenneth O Stanley.  
Compositional pattern producing networks: A novel abstraction of development.  
*Genetic programming and evolvable machines*, 8(2):131–162, 2007.