Framsticks model and genetics

Maciej Komosinski Szymon Ulatowski

www.framsticks.com

Outline

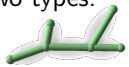
- organism model
  - body
  - brain
  - sensors and effectors
- genetics
  - representations
  - conversions
  - operators

Organism. Elements

- body
  - Parts
  - Joints
- brain
  - Neurons
    - signal processing / sensors / actuators
    - embodied or not
  - Connections

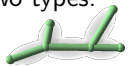
Organism. Body elements

The model can be of one of the two types:

SHAPETYPE_BALL_AND_STICK  or SHAPETYPE_SOLIDS 

Organism. Body elements

The model can be of one of the two types:

SHAPETYPE_BALL_AND_STICK  or SHAPETYPE_SOLIDS 

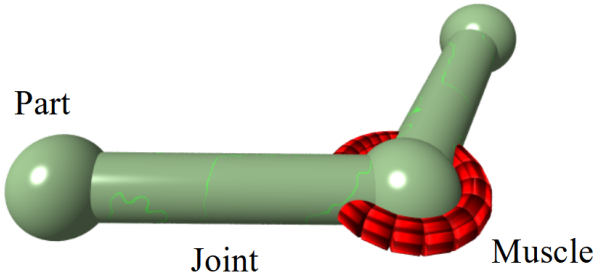
- Parts
  - type
    - for SHAPETYPE_BALL_AND_STICK – a point,
    - for SHAPETYPE_SOLIDS – ellipsoid/box/cylinder
  - 3D position
  - 3D orientation
  - physical properties: mass, friction, etc.
  - experiment-specific properties: ingestion and assimilation ability, ...
- Joints
  - references of the two Parts
  - can be "relative" (store information about length, and set coordinates of the other Part wrt. the first Part)
  - physical properties: axial stiffness, rotational stiffness, etc.
  - experiment-specific properties: stamina, ...

Organism. Body constraints

- at most one Joint can directly connect two Parts
- each Joint must be connected with (must be incident on) two distinct Parts
- all Parts must be directly or indirectly connected with each other
- relative Joints must not form cycles

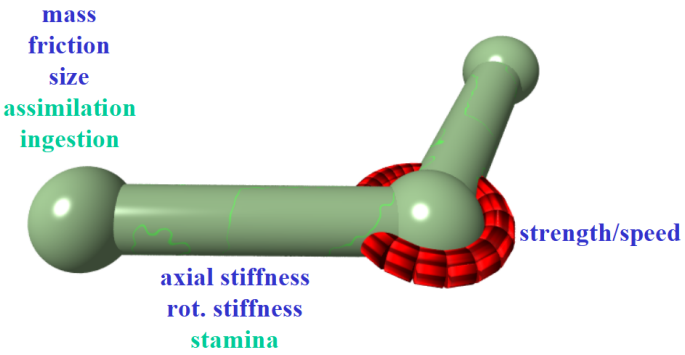
Organism. Body properties

Physical	Experiment-specific examples
• Parts: mass, friction, size • Joints: axial stiffness, rotational stiffness • Muscles: strength/speed	• Parts: assimilation, ingestion • Joints: stamina • Muscles: energy consumption



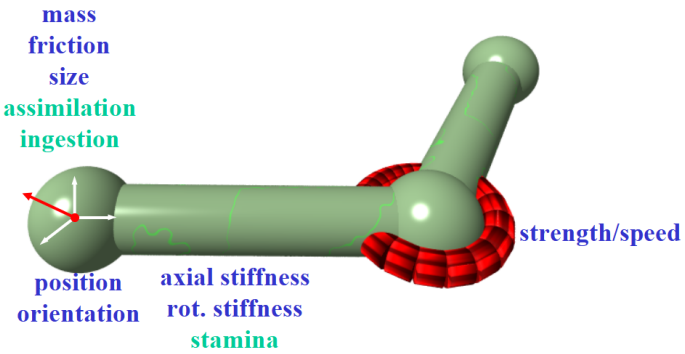
Organism. Body properties

Physical	Experiment-specific examples
• Parts: mass, friction, size • Joints: axial stiffness, rotational stiffness • Muscles: strength/speed	• Parts: assimilation, ingestion • Joints: stamina • Muscles: energy consumption



Organism. Body properties

Physical	Experiment-specific examples
• Parts: mass, friction, size • Joints: axial stiffness, rotational stiffness • Muscles: strength/speed	• Parts: assimilation, ingestion • Joints: stamina • Muscles: energy consumption



Organism. Brain

- any topology of a neural network, synchronous update
- neurons embodied (Parts, Joints) or not
- implement any function
- inputs: none / one / many
- outputs: none / one (may have many channels)
- a list of neural properties (parameters)
- definition: C++ or a script (`*.neuro` file)
- weighted connections

Organism. Brain. Neuron example

Short name: Thr
Long name: Threshold

- single input
- single output
- properties:
  - t (threshold)
  - hi (high output value)
  - lo (low output value)
- if (input$\geq$t) then output:=hi else output:=lo

Organism. Brain. Neuron example

Short name: N
Long name: Neuron

- many inputs
- single output
- properties:
  - fo (force)
  - in (intertia)
  - si (sigmoid)

$$o_t = \frac{2}{1+e^{-s_t \cdot Sigmoid}} - 1$$

$$s_t = s_{t-1} + v_t$$

$$v_t = v_{t-1} \cdot interia + force \cdot (i_t - s_{t-1})$$

i – weighted sum of inputs
v – speed of changes
s – internal state
o – neuron output
Subscript t is the moment of time.

force := 1
inertia := 0

$$O_t = \frac{2}{1 + e^{-i_t \cdot Sigmoid}} - 1$$

(note that in this case, s_t becomes i_t)

Organism. Brain. Neuron example

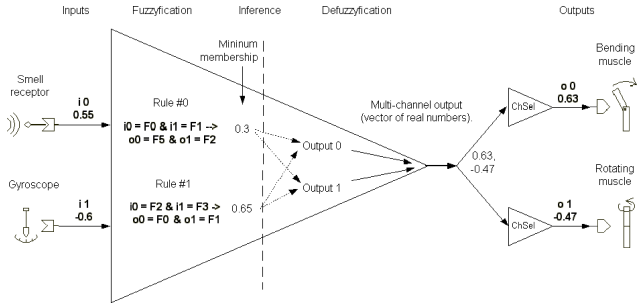
- Short name: Fuzzy
- Long name: Fuzzy neuron

- many inputs
- single output (with many channels)
- properties:
  - fuzzy sets
  - fuzzy rules
- represents a fuzzy rule-based system [HKW03; HK08]

Organism. Brain. Neuron example

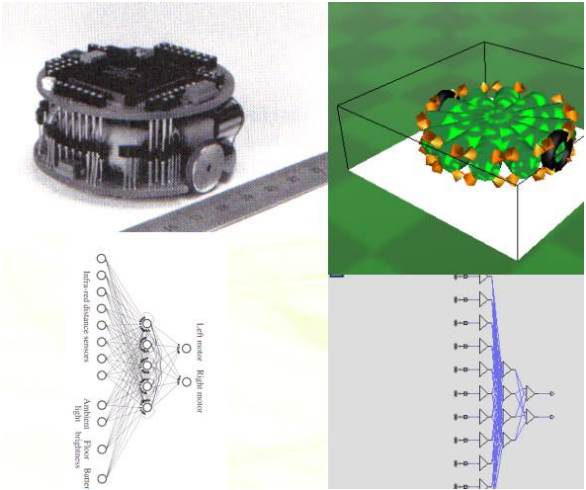
- Short name: Fuzzy
- Long name: Fuzzy neuron

- many inputs
- single output (with many channels)
- properties:
  - fuzzy sets
  - fuzzy rules
- represents a fuzzy rule-based system [HKW03; HK08]



Organism. Brain. Neuron example

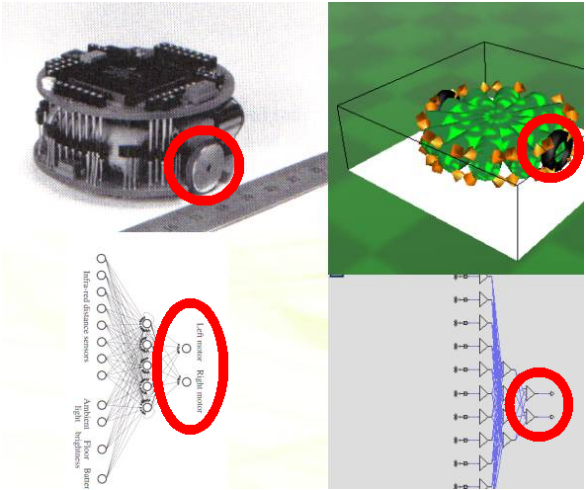
A custom "Wheel" effector for robotic experiments



- affects movement of a Part in the agent

Organism. Brain. Neuron example

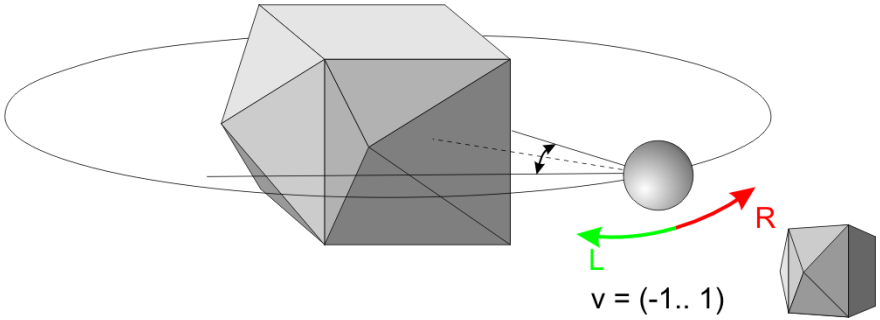
A custom "Wheel" effector for robotic experiments



- affects movement of a Part in the agent

A vector eye (VEye) sensor

- optional input controls tilt (rotation)
- single output (with many channels) outputs perceived vector coordinates
- properties: the object (sic!), scale, perspective
- details: [JK06], video

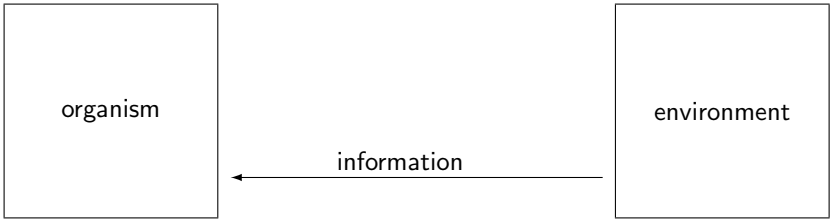


Organism. Brain. Neuron list

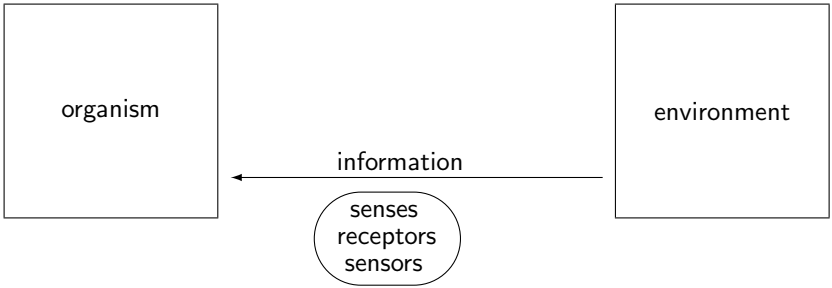
Short name	Long name	Description
N	Neuron	Standard Framsticks sigmoid neuron
G	Gyroscope	Tilt sensor
T	Touch	Touch sensor
S	Smell	Smell sensor
*	Const	Constant value
|	Bend muscle	
@	Rotation muscle	
D	Differentiate	Calculate the difference between the current and previous input value
Ch	Channelize	Combines all input signals into single multichannel output
ChMux	Channel multiplexer	Outputs one channel from first (multichannel) first input, selected by the second
ChSel	Channel selector	Output one channel from multichannel input, selected by the "ch" parameter
Rnd	Random value	
Sin	Sinus Generator	Output frequency = f0 + input
Delay	Delay	
Thr	Threshold	if (input>=t) then output=hi else output=lo
Fuzzy	Fuzzy neuron	
VEye	Vector eye	
LMu	Length muscle	
Water	Water detector	
Energy	Energy level	

Organism. Simulation. Interactions

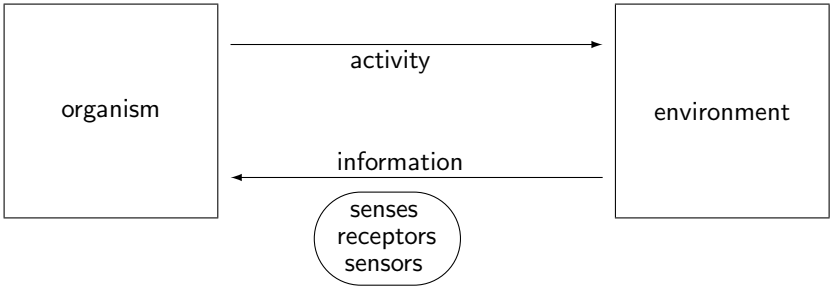
Organism. Simulation. Interactions



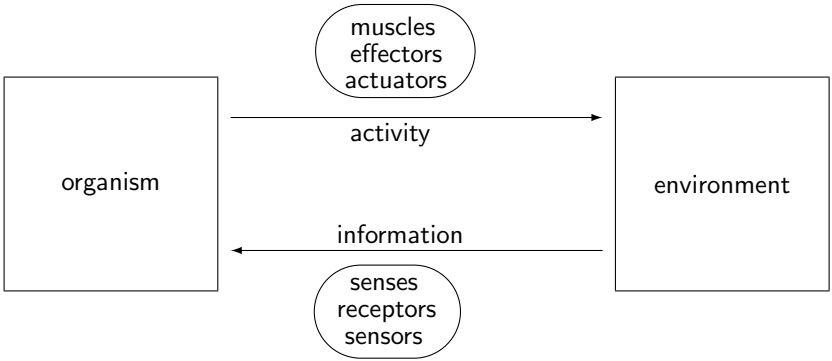
Organism. Simulation. Interactions



Organism. Simulation. Interactions

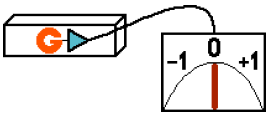


Organism. Simulation. Interactions

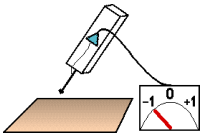


Receptors and effectors

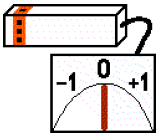
tilt



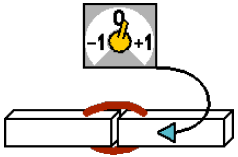
touch

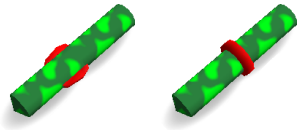


smell



bending and rotating muscles

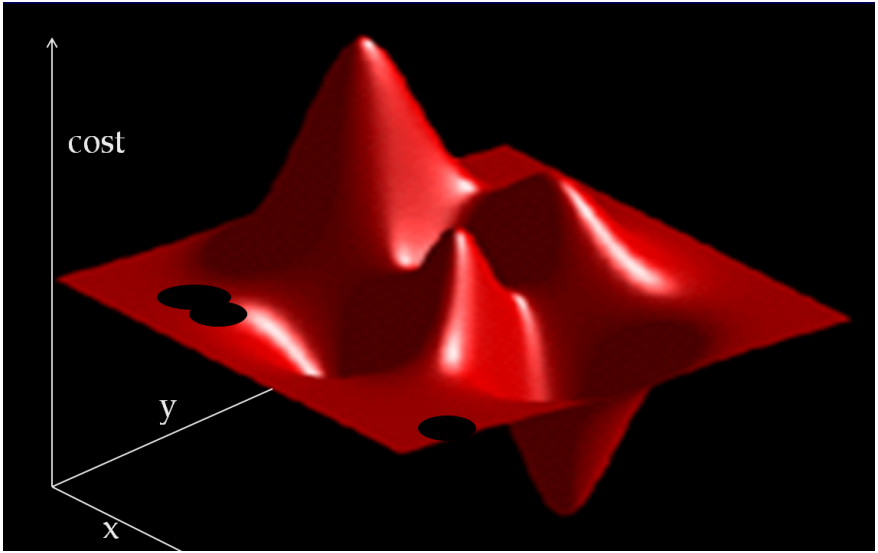




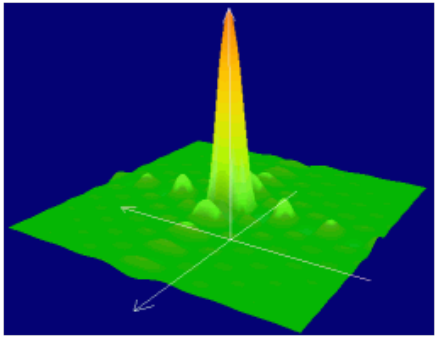
DNA ⟶ organism 

genotype ⇉ model

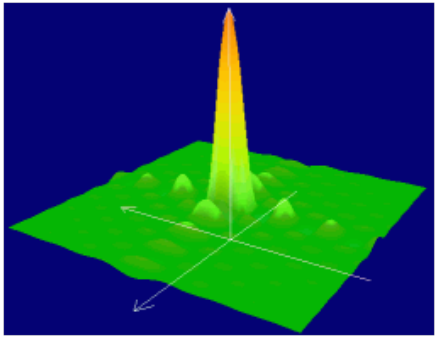
Why so important? . . . Fitness landscapes!

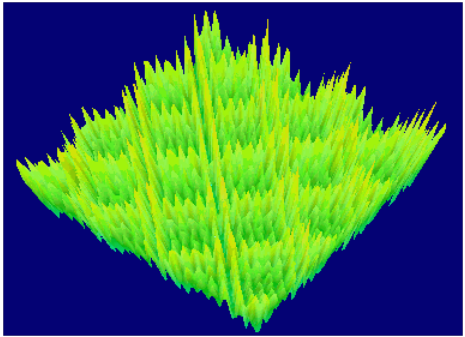


Why so important? . . . Fitness landscapes!

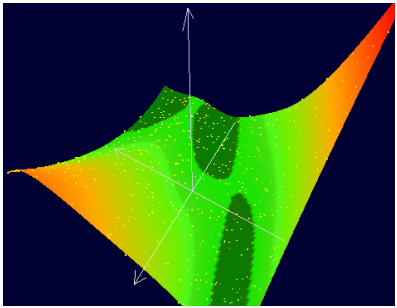


Why so important? . . . Fitness landscapes!

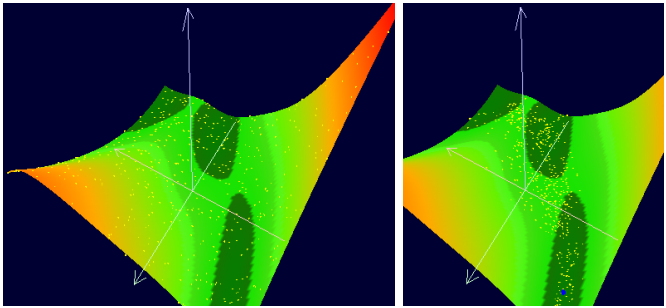




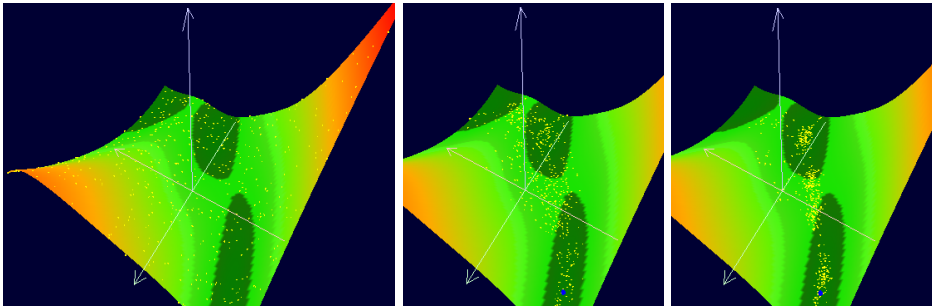
Stages of search (lower = better):



Stages of search (lower = better):

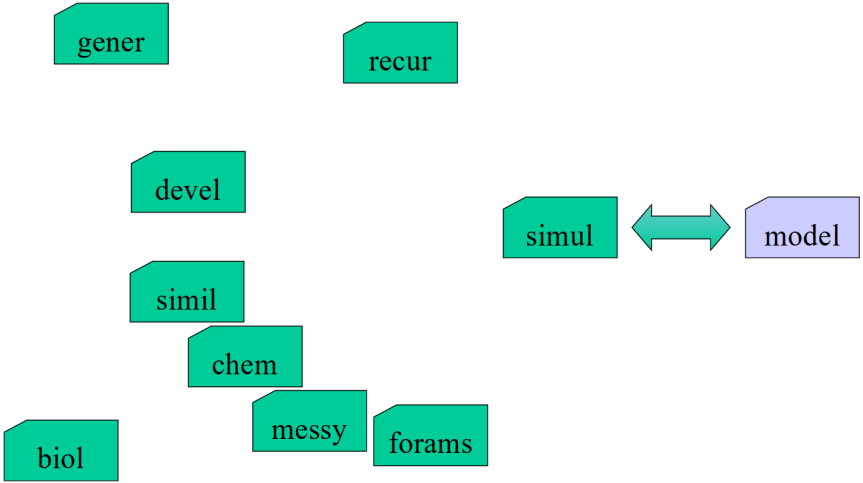


Stages of search (lower = better):



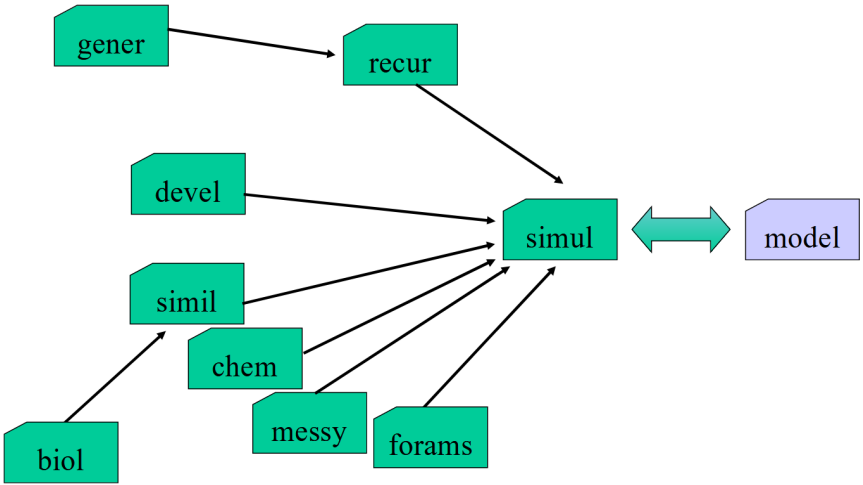
Genetic encoding conversion graph

[online]



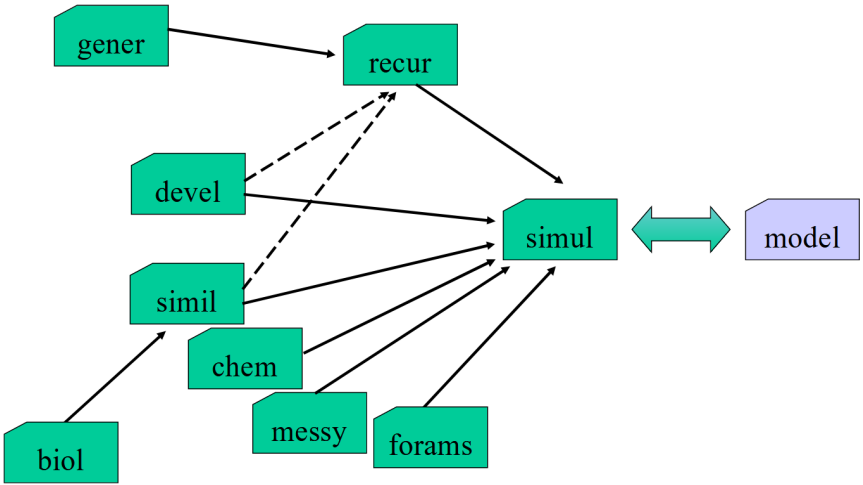
Genetic encoding conversion graph

[online]



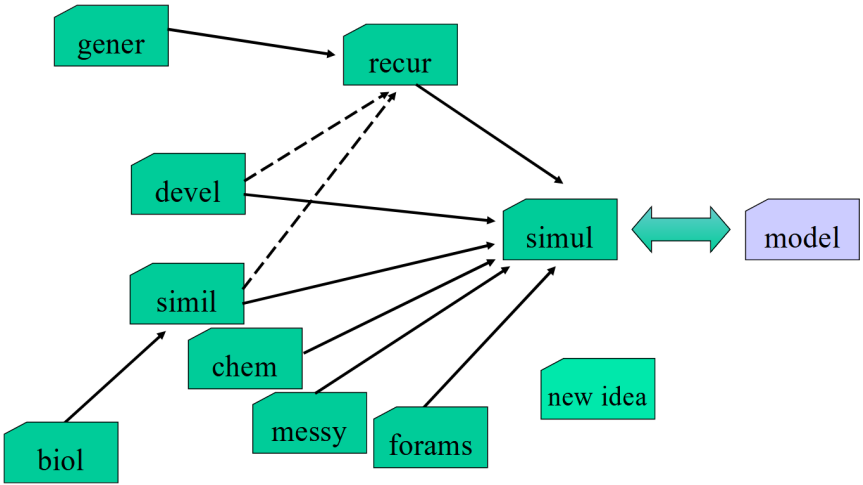
Genetic encoding conversion graph

[online]



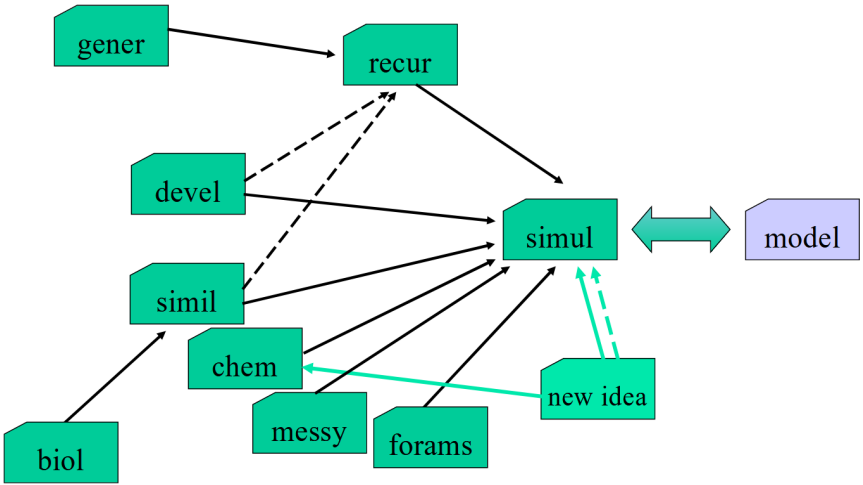
Genetic encoding conversion graph

[online]



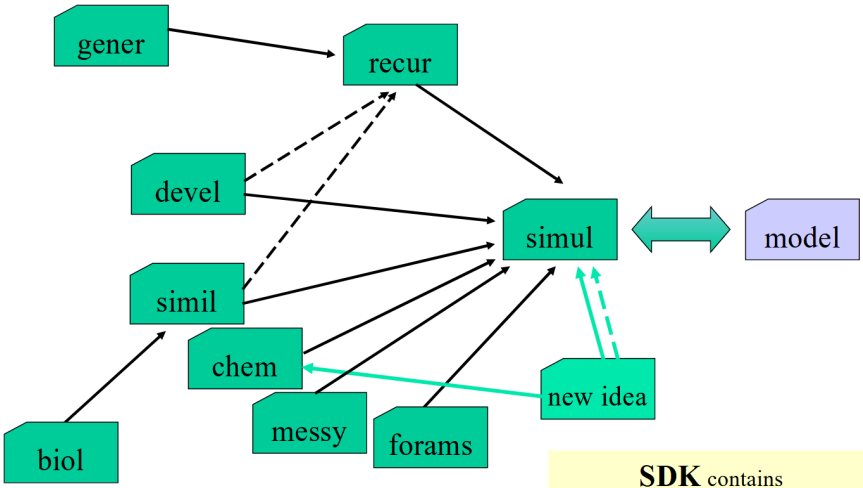
Genetic encoding conversion graph

[online]



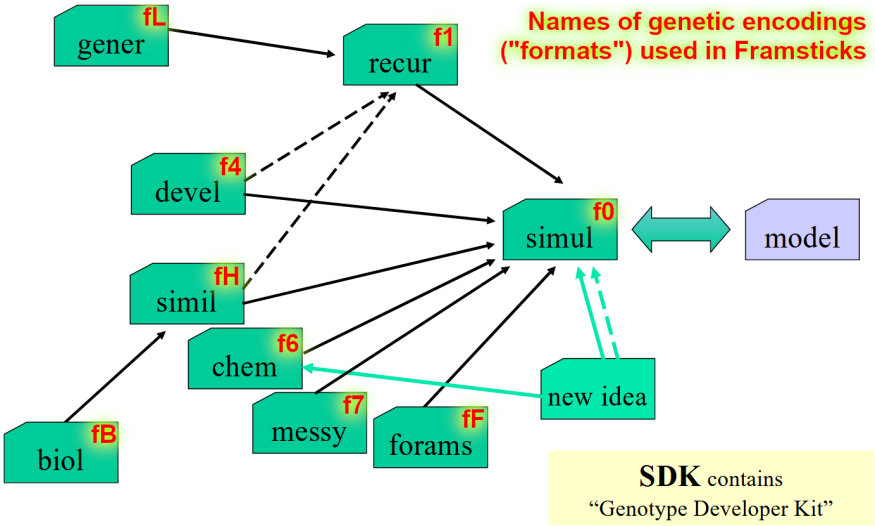
Genetic encoding conversion graph

[online]

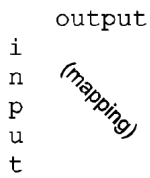


Genetic encoding conversion graph

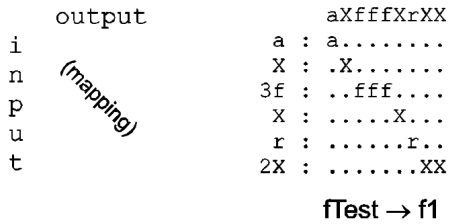
[online]



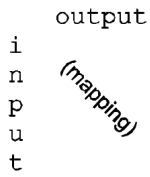
Genetics. Mapping gene ⇔ gene ⇔ . . . ⇔ phene. [KU04]



Genetics. Mapping gene ⇔ gene ⇔ . . . ⇔ phene. [KU04]



Genetics. Mapping gene ⇔ gene ⇔ . . . ⇔ phene. [KU04]




```
      aXfffXrXX
 a : a........
 X : .X.......
3f : ..fff....
 X : .....X...
 r : ......r..
2X : .......XX
```

fTest → f1

```
      Parts      Joints     Neurons
      012345678901234567890123456789
  c : ..............................
  X : 01........0.....................
  [ : ....................0.........
  | : ....................01........
1:1] : ....................0.........
  X : ..2........1...................
[G] : ......................2.......
```

f1 → phenotype

Genetics. Mapping gene ⇔ gene ⇔ . . . ⇔ phene. [KU04]

output (mapping)

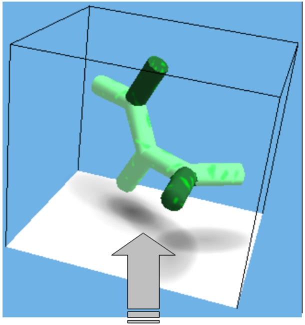
input

		aXfffXrXX
a	:	a........
X	:	.X.......
3f	:	..fff....
X	:	.....X...
r	:	......r..
2X	:	XX

fTest → f1

		Parts Joints Neurons
		012345678901234567890123456789
c	:	....
X	:	01........0...................
[	:	...................0..........
|	:	...................01.........
1:1]	:	...................0..........
X	:	..2........1..................
[G]	:	.....................2........

f1 → phenotype



X(X,RRX(X,X(X,X)))

Genetics. Mapping gene ⇔ gene ⇔ . . . ⇔ phene. [KU04]

input

(mapping)

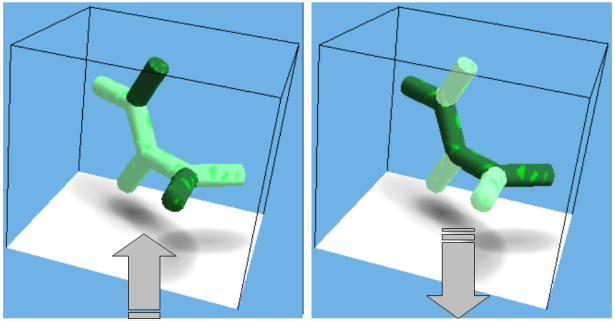
output

```
          aXfffXrXX
a  :  a........
X  :  .X.......
3f :  ..fff....
X  :  .....X...
r  :  ......r..
2X :  .......XX
```

```
          Parts      Joints     Neurons
          012345678901234567890123456789
c   :  ..............................
X   :  01........0.....................
[   :  ......................0........
|   :  ......................01.......
1:1] :  ......................0........
X   :  ..2........1...................
[G] :  ......................2.......
```

fTest → f1

f1 → phenotype

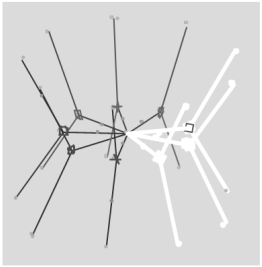


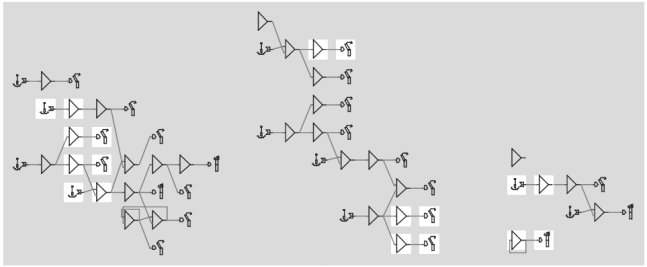
X(X,RRX(X,X(X,X)))

X(X,RRX(X,X(X,X)))

Genetics. Mapping gene ⇔ gene ⇔ . . . ⇔ phene. [KU04]

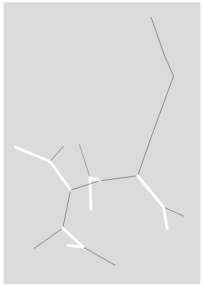
(,LFMW(LLX[| 1 :3.455,0 :1.453][| 1 :-509.744,-1 :1.033][@9 :-1] ,,rrX[/ :2.834]a(rrX[G:0.827](RLLX[|9:-1], LLiqwX[@G :93.351,-1 :727.177]),rrfMX[G:1](LLcMX[@0 :2.095][|8:-1], LLfX[|7 :1]),rrX[G:1.151, -1 :1.629]F(LLcfX[|8:-0.859], LLX[|7:1.482]), rrX[G:3.254]F(LLX[|9:-2.621], LLX[|8 :0.783]),rrX[G:1]F(LLwX [|9:-1],LLX[|1 :1.221,-1 :-3.208]), rrwX[G:1.052](LLFiX[@ 1 :1.601] [|-20:-0.665],LLQX[! :1.000]), rrX[G:0.757,-1 :-2.644]f(rLLMMX[|-21:-1.377], LLX[|-22 :0.984]), rrQX[G:0.887,-1 :0.750](LLX [|G:-0.930],LLX[|-22 :2.345])) ,,LLX[|-19:-4.840,-1 :-0.757]))

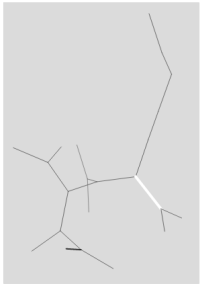




Genetics. Mapping gene ⇔ gene ⇔ . . . ⇔ phene. [KU04]

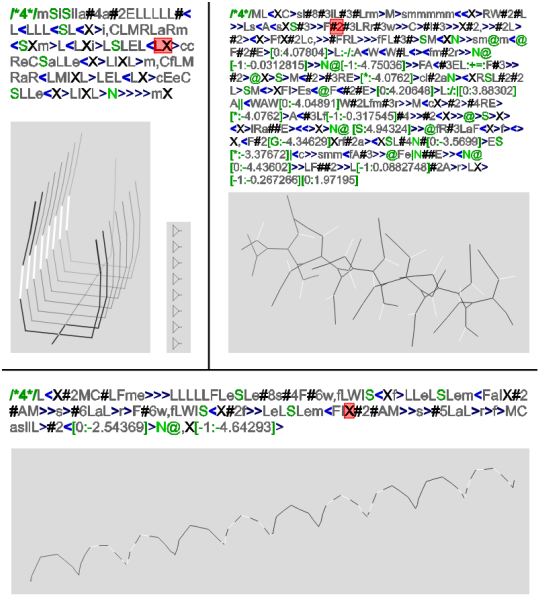
/*4*/Amm#4LL<<LX#2#3
WRe>#4ER#5ER>>>W#2
>><<X#3#2R>#2ML#2>F
>>#3#2E>>f>,RA#3W#4#
2R>#2ML#2>F>>#2>>RA
X#3#R>c>E>X#R#2RL>>
EcCM>aF#4SML#2>>>LL
<#3#2R>#4ML#2R>F>>#
3#2E>>X><X><#3R>RA
X#3#2RL>c#2M>>E><IE
cCMX>X#4ML>





/*4*/Amm#4LL<<LX#2#3
WRe>#4ER#5ER>>>W#2
>><<X#3#2R>#2ML#2>F
>>#3#2E>>f>,RA#3W#4#
2R>#2ML#2>F>>#2>>RA
X#3#R>c>E>X#R#2RL>>
EcCM>aF#4SML#2>>>LL
<#3#2R>#4ML#2R>F>>#
3#2E>>X><X><#3R>RA
X#3#2RL>c#2M>>E><IE
cCMX>X#4ML>

Genetics. Mapping gene ⇔ gene ⇔ . . . ⇔ phene. [KU04]



Characteristics of genetic representations [KR01]

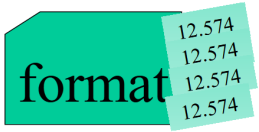
	Complexity		Constraints	
	Genotype	Interpretation	Body	Brain
simul	Med	Low	None	None
recur	Med	Med	High	Low
devel	High	Med	High	Low

	Modularity	Symmetry	Compression	Redundancy
simul	None	None	None	None
recur	None	Low	None	Low
devel	High	High	Var	None

Characteristics of genetic representations

	Complexity		*Constraints*		*Cyclic Body*	*Compression*	*Redundancy*
	Format	*Interpret*	*Body*	*Brain*			
simul	Med	Low	None	None	Y	None	None
recur	Med	Med	High	Low	N	None	Low
simil	Low	High	Med	None	Y/N	Low	None
chem	Low	High	Med	None	Y/N	Var	Var
devel	High	Med	High	Low	N	Var	None
messy	High	Low	High	?	N	None	None
...							

Genetic operators



Genetic operators

Mutation

Crossover

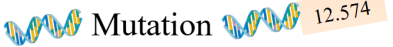
Repair

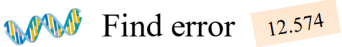
Find error

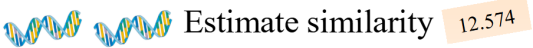
Estimate similarity

Simplify

Genetic operators







f0 representation

- all elements directly described
- basic, internal format
- "serialization" of a Model
- supports geometric relativity

f0 representation

- all elements directly described
- basic, internal format
- "serialization" of a Model
- supports geometric relativity

```
//0
p:
p:1.0
p:1.5,−0.612,0.612
p:1.5,0.612,−0.612
j:0,1,rx=−0.7854,dx=1.0,0.0,0.0
j:1,2,rx=−0.5184,rz=−1.0472,dx=1.0,0.0,0.0
j:1,3,rx=−0.5184,rz=1.0472,dx=1.0,0.0,0.0
n:j=1,d=@:p=0.25
n:p=3,d=Sin
c:0,1
```

f0 representation

- all elements directly described
- basic, internal format
- "serialization" of a Model
- supports geometric relativity

```
//0
p:
p:1.0
p:1.5,−0.612,0.612
p:1.5,0.612,−0.612
j:0,1,rx=−0.7854,dx=1.0,0.0,0.0
j:1,2,rx=−0.5184,rz=−1.0472,dx=1.0,0.0,0.0
j:1,3,rx=−0.5184,rz=1.0472,dx=1.0,0.0,0.0
n:j=1,d=@:p=0.25
n:p=3,d=Sin
c:0,1
```

Equivalent to this **f1** genotype:

qX(X[@,1:1],X[Sin])

which was converted to **f0** according to the genetic encoding conversion graph.

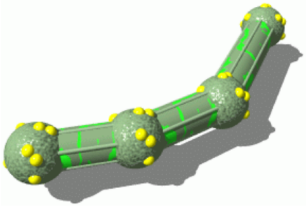
f0 genotype–phenotype relation

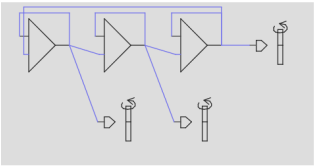

```
//0
p:fr=0.025, vg=0.875
p:0.351, fr=0.025, vg=0.875
p:0.245, 0.324, fr=0.0062, vg=0.875
p:–0.195, 0.397, fr=0.1, vg=0.875
...
j:0, 1, dx=0.351, 0.0, 0.0
j:1, 2, rz=1.884, dx=0.341, 0.0, 0.0
j:1, 3, rz=2.513, dx=0.675, 0.0, 0.0
j:3, 4, rx=0.785, rz=–1.5, dx=0.393, 0.0, 0.0
...
n:j=2, d=@:p=0.625
n:p=4, d=N:in=0.0
n:j=3, d="|:p=0.55,r=0.333333"
...
c:0, 2, 1.272
c:1, 0
c:2, 8, 0.931
...
```

f0 genotype–phenotype relation


```
//0
parts
p:fr=0.025, vg=0.875
p:0.351, fr=0.025, vg=0.875
p:0.245, 0.324, fr=0.0062, vg=0.875
p:–0.195, 0.397, fr=0.1, vg=0.875
...
joints
j:0, 1, dx=0.351, 0.0, 0.0
j:1, 2, rz=1.884, dx=0.341, 0.0, 0.0
j:1, 3, rz=2.513, dx=0.675, 0.0, 0.0
j:3, 4, rx=0.785, rz=–1.5, dx=0.393, 0.0, 0.0
...
neurons
n:j=2, d=@:p=0.625
n:p=4, d=N:in=0.0
n:j=3, d="|:p=0.55,r=0.333333"
...
conn's
c:0, 2, 1.272
c:1, 0
c:2, 8, 0.931
...
```

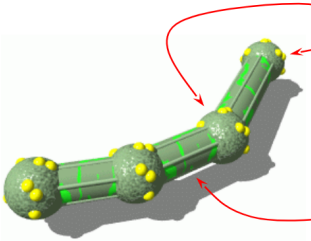
f0 genotype–phenotype relation





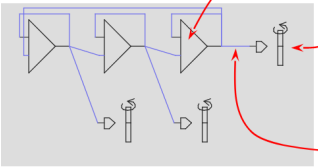
```
//0
parts
p:fr=0.025, vg=0.875
p:0.351, fr=0.025, vg=0.875
p:0.245, 0.324, fr=0.0062, vg=0.875
p:–0.195, 0.397, fr=0.1, vg=0.875
...
joints
j:0, 1, dx=0.351, 0.0, 0.0
j:1, 2, rz=1.884, dx=0.341, 0.0, 0.0
j:1, 3, rz=2.513, dx=0.675, 0.0, 0.0
j:3, 4, rx=0.785, rz=–1.5, dx=0.393, 0.0, 0.0
...
neurons
n:j=2, d=@:p=0.625
n:p=4, d=N:in=0.0
n:j=3, d="|:p=0.55,r=0.333333"
...
conn's
c:0, 2, 1.272
c:1, 0
c:2, 8, 0.931
...
```

f0 genotype–phenotype relation



parts

joints

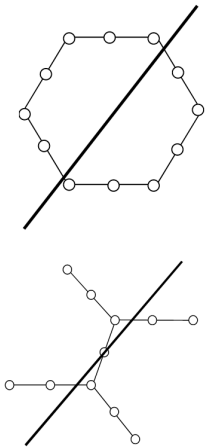


neurons

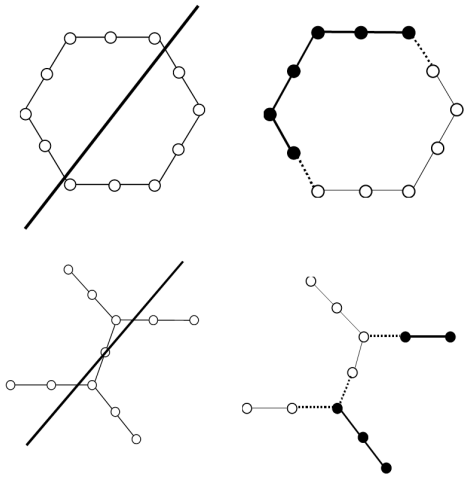
conn's

```
//0
p:fr=0.025, vg=0.875
p:0.351, fr=0.025, vg=0.875
p:0.245, 0.324, fr=0.0062, vg=0.875
p:–0.195, 0.397, fr=0.1, vg=0.875
...
j:0, 1, dx=0.351, 0.0, 0.0
j:1, 2, rz=1.884, dx=0.341, 0.0, 0.0
j:1, 3, rz=2.513, dx=0.675, 0.0, 0.0
j:3, 4, rx=0.785, rz=–1.5, dx=0.393, 0.0, 0.0
...
n:j=2, d=@:p=0.625
n:p=4, d=N:in=0.0
n:j=3, d="|:p=0.55,r=0.333333"
...
c:0, 2, 1.272
c:1, 0
c:2, 8, 0.931
...
```

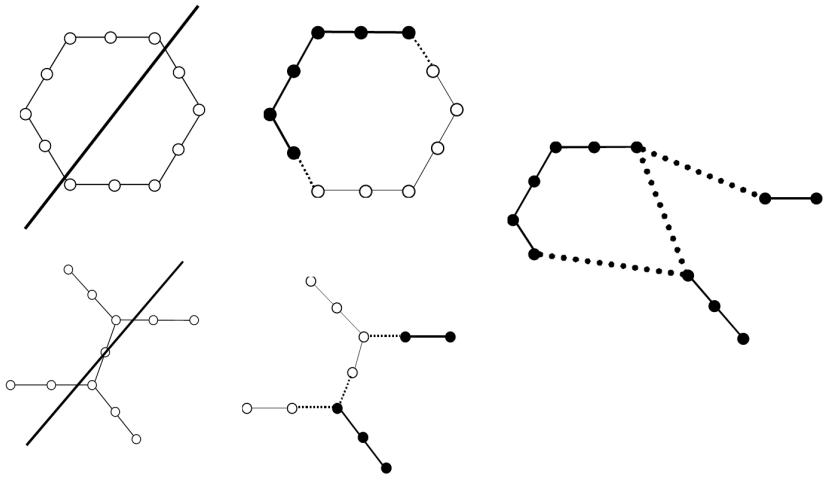
f0 crossing-over: idea



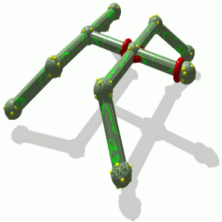
f0 crossing-over: idea



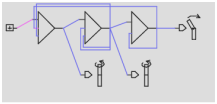
f0 crossing-over: idea



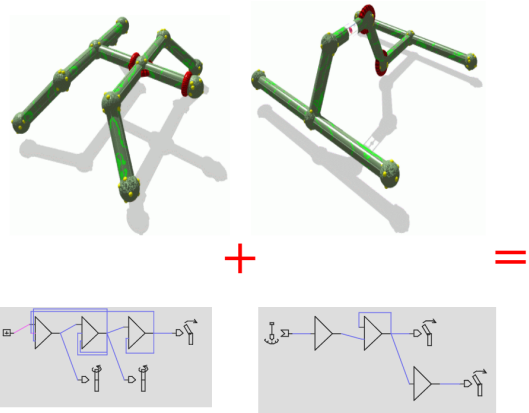
f0 crossing-over: example



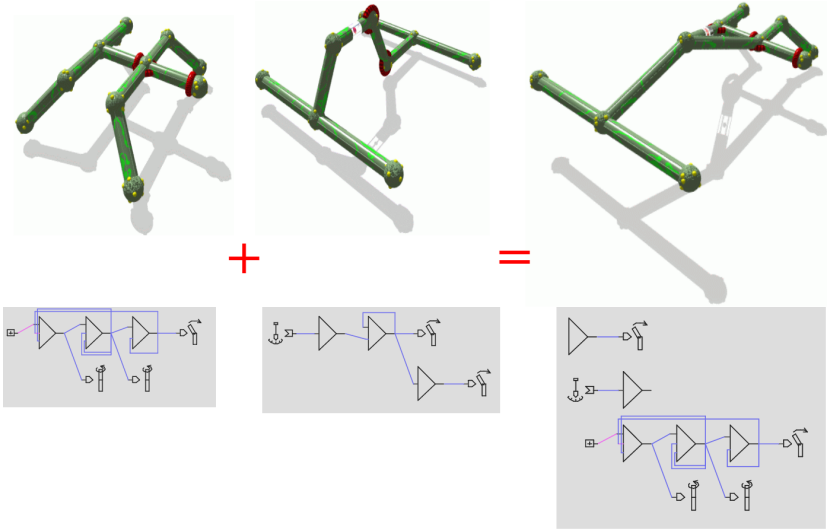
+



f0 crossing-over: example



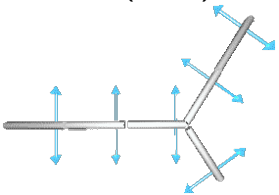
f0 crossing-over: example



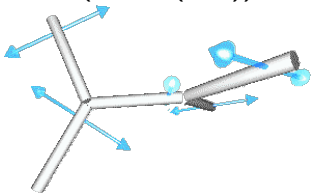
f1 representation

- properties are local, relative
- properties propagate along the body
- control elements (neurons, sensors) are near elements under control (muscles, sticks)
- recursive body (tree)
- any topology of NN
- human-friendly

XXX(XX,X)

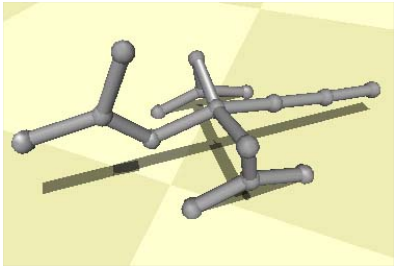


X(X,RRX(X,X))

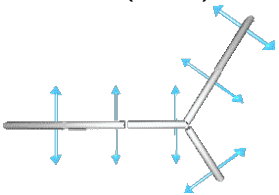


f1 representation

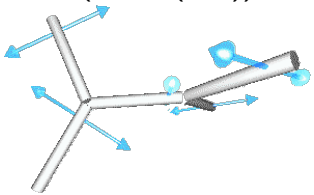
- properties are local, relative
- properties propagate along the body
- control elements (neurons, sensors) are near elements under control (muscles, sticks)
- recursive body (tree)
- any topology of NN
- human-friendly



XXX(XX,X)

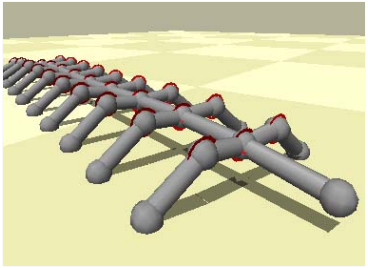


X(X,RRX(X,X))

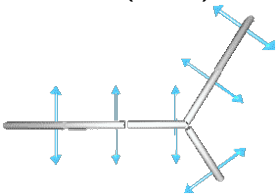


f1 representation

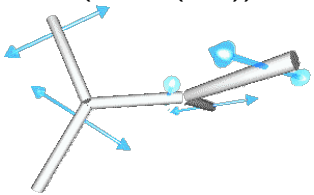
- properties are local, relative
- properties propagate along the body
- control elements (neurons, sensors) are near elements under control (muscles, sticks)
- recursive body (tree)
- any topology of NN
- human-friendly



XXX(XX,X)

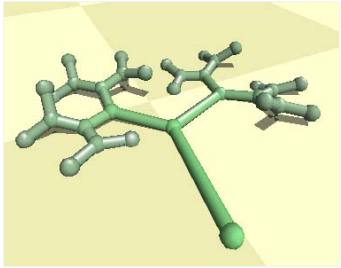


X(X,RRX(X,X))

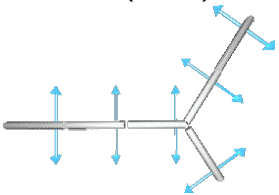


f1 representation

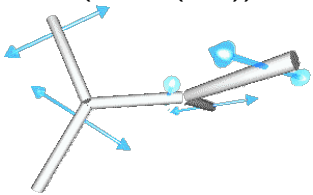
- properties are local, relative
- properties propagate along the body
- control elements (neurons, sensors) are near elements under control (muscles, sticks)
- recursive body (tree)
- any topology of NN
- human-friendly



XXX(XX,X)



X(X,RRX(X,X))



f1 “modifiers”

R r	Rotation of the branching plane by 45°
Q q	Twist of the branching plane
C c	Curvedness
L l	Length
F f	Friction
M m	Muscle strength

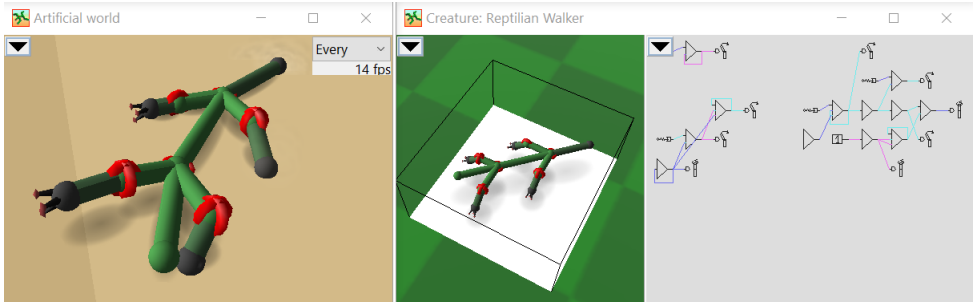
A complete description: https://www.framsticks.com/a/al_geno_f1.html

f1 example

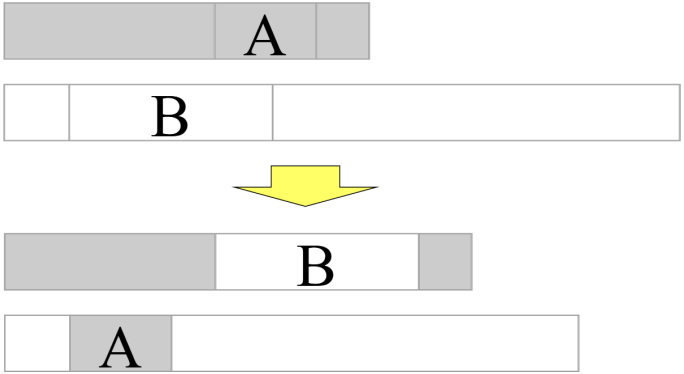
db(,rrlMMMMXlFFFFCgX[|T:10.159,/:−1.442,1:3.562][@0:−51.595],FFFFlL
X[|0:2.744,−2:−3.181,−1:1.151][8:2.682],rrMMXlFFFFMMMMCgX[|T:−162.1
72,−1:8.977][@4:−0.573,3:0.724,fo:1],,,LLLXMMM(rrlMXlFFFFCgX[|T:−80.858,0:
4.784][@*:8.62],,,gX[0:657.704,−1:−3.466,−1:−346.898][|−6:2.895,fo:0.208],,,rrlMXl
FFFFCgX[N,si:999][|T:−78.873,0:2.585,−1:−2.867]))

f1 example

db(,rrlMMMMXIFFFFCgX[|T:10.159,/:−1.442,1:3.562][@0:−51.595],FFFFIL X[|0:2.744,−2:−3.181,−1:1.151][8:2.682],rrMMXIFFFFMMMMCgX[|T:−162.1 72,−1:8.977][@4:−0.573,3:0.724,fo:1],,,LLLXMMM(rrlMXIFFFFCgX[|T:−80.858,0: 4.784][@*:8.62],,,gX[0:657.704,−1:−3.466,−1:−346.898][|−6:2.895,fo:0.208],,,rrlMXI FFFFCgX[N,si:999][|T:−78.873,0:2.585,−1:−2.867]))



f1 crossing-over



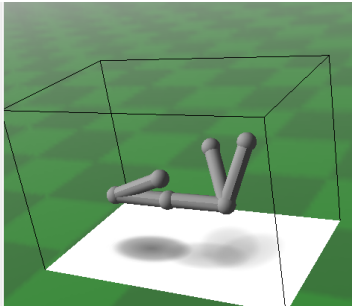
Cutpoints may be selected proportionally to the length of both parents.

fL representation

- A parametric generative Lindenmayer system
- A set of production rules with parameters
- Rules are activated and generate a genotype
- For example:

Genotype

```
//L
w:w0, 2
w:w1
w:w2, 1
i:axiom=C(-0.7267972738482058)C(0.6512542888522148)w0()
[[rotZ(0.01436303136870265)S()[S()[rotY(0.47939520375803113)
rotZ(1.7156715150922537)S()]rotY(0.6546807433478534)
rotZ(0.7289054011926055)S(0.6272945767268538)]]
rotY(0.24517498910427094)S()]S(rotstif=0.9601573273539543), 0.0,
maxwords=300
r:pred=w1, succ=C(0.33147175842896104)
r:pred=w2, cond=$0=0.6200042264536023&$0>0.5498812331352383,
succ=S(0.5446961037814617)
r:pred=w2, succ=S()
```



fH representation

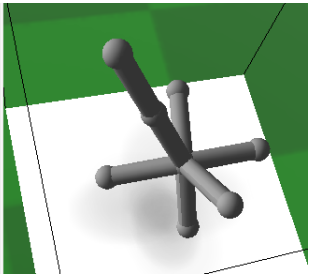
- Body:
  - list of body components (sticks) with "links" and properties
  - joined according to links' similarity
- Brain:
  - list of NN connections, effectors, sensors with "links" and properties
  - connected according to links' similarity

fH representation

- Body:
  - list of body components (sticks) with "links" and properties
  - joined according to links' similarity
- Brain:
  - list of NN connections, effectors, sensors with "links" and properties
  - connected according to links' similarity

Genotype

```
//H
3
j:0.1912, 0.509, 0.4535, -0.9480, -0.9568, 0.4396, fr=0.221
j:fr=0.619
j:0.1912, 0.5090, 0.4535, -0.9481, -0.9568, 0.4396, fr=0.221
j:-0.6167, 0.3991, -0.5147, 0.2721, 0.6604, -0.9171, l=1.055
j:fr=0.619
j:0.1912, 0.5091, 0.4535, -0.9481, -0.9568, 0.4396, fr=0.221
j:0.6999, 0.4073, -0.7687, -0.1117, -0.8154, 0.4741, rotstif=0.88
```



fB representation

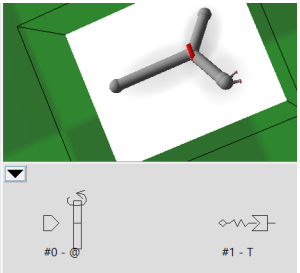
- 26 characters of latin alphabet, from ‘a’ to ‘z’
- every sequence starting after **aa** and extending to the first **zz** is considered a gene
- neurons are encoded in quotation marks using their original names
- genes are interpreted as encoded objects in the **fH** encoding (i.e., **fB** is converted to **fH** as shown earlier in the genetic encoding conversion graph)
- this encoding exhibits properties similar to DNA
- Operators: horizontal gene transfer, crossing over, substitution, deletion, insertion, gene duplication, translocation

fB representation

- 26 characters of latin alphabet, from ‘a’ to ‘z’
- every sequence starting after **aa** and extending to the first **zz** is considered a gene
- neurons are encoded in quotation marks using their original names
- genes are interpreted as encoded objects in the **fH** encoding (i.e., **fB** is converted to **fH** as shown earlier in the genetic encoding conversion graph)
- this encoding exhibits properties similar to DNA
- Operators: horizontal gene transfer, crossing over, substitution, deletion, insertion, gene duplication, translocation

Genotype

```
//B
3
aah"@:p=0.97"faaktj"T:r=0.3"bwgydajsdaabnzz
```



f6 representation

- “chemical” substances in 3D
- transforms initial substances into an organism
- rules of growth of body and brain
- saturation threshold to fire a rule
- propagation and changes of substances along growth directions
- propagation of properties of grown elements

f6 representation

- "chemical" substances in 3D
- transforms initial substances into an organism
- rules of growth of body and brain
- saturation threshold to fire a rule
- propagation and changes of substances along growth directions
- propagation of properties of grown elements
- example:

4 rules
3 substances
2 properties

0.144 0.833 0.940, 0.546 0.249 grow stick, 0.859 0.604 0.707, 0.516 0.600
0.941 0.876 0.303, 0.038 0.630 grow stick, 0.902 0.320 0.035, 0.648 0.525
0.767 0.201 0.636, 0.751 0.022 grow stick, 0.321 0.661 0.663, 0.311 0.319
0.951 0.283 0.454, 0.428 0.997 grow stick, 0.996 0.554 0.162, 0.192 0.160

f7 representation

- "messy" encoding
- any string of uppercase characters is a valid genotype
- simple genetic operators
- various interpretation approaches are possible, for example:
  - sections correspond to elements of body and brain:
    Z AAAAA BCLQU BCLQU BCLQU YYYYB BCNDG BCLQU BCLQU ...
  - SectionTag 'Z' starts the Parts section:
    AAAAA – label, BCLQU BCLQU BCLQU ... – coordinates
  - labels are calculated as follows:
    $\mathrm{AAAAA} = 0 \cdot 26^4 + 0 \cdot 26^3 + 0 \cdot 26^2 + 0 \cdot 26^1 + 0 \cdot 26^0 = 0$
    $\mathrm{YYYYB} = 24 \cdot 26^4 + 24 \cdot 26^3 + 24 \cdot 26^2 + 24 \cdot 26^1 + 1 \cdot 26^0 = 11406097$
    ...

f4 representation

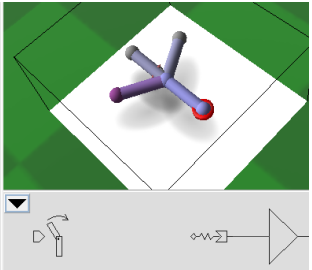
- encodes development of “cells” (division and differentiation)
- genes are commands of differentiation
- these instructions are executed in parallel
- supports symmetry and modularity
- development starts with a single, undifferentiated ancestor cell
- stops when all the cells are differentiated
- a complete description: https://www.framsticks.com/a/al_geno_f4.html

f4 representation

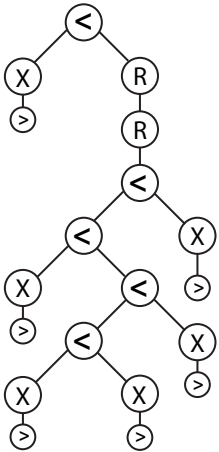
- encodes development of “cells” (division and differentiation)
- genes are commands of differentiation
- these instructions are executed in parallel
- supports symmetry and modularity
- development starts with a single, undifferentiated ancestor cell
- stops when all the cells are differentiated
- a complete description: https://www.framsticks.com/a/al_geno_f4.html

Genotype

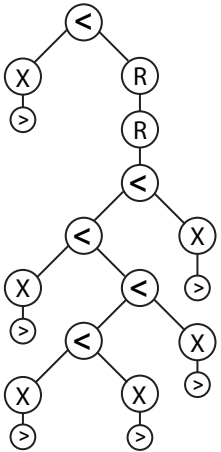
```
/*4*/<<<<<BX><N:N[-2:-3.12]>X>gX>N:|>Xc>
<N:T>X>
```



f4 development



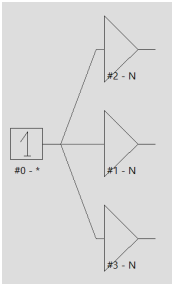
f4 development



/*4*/<X>RR<<X><<<X>X>X>X

f4 examples

Neural links are duplicated when a neuron divides:

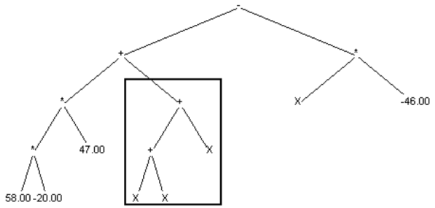


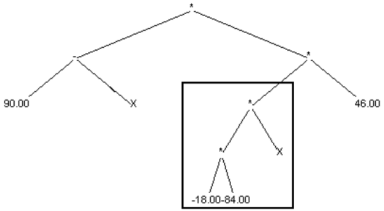
/*4*/<X><N:*>
N:N[−1:10]<><>>

Parts of the genotype may be interpreted many times:

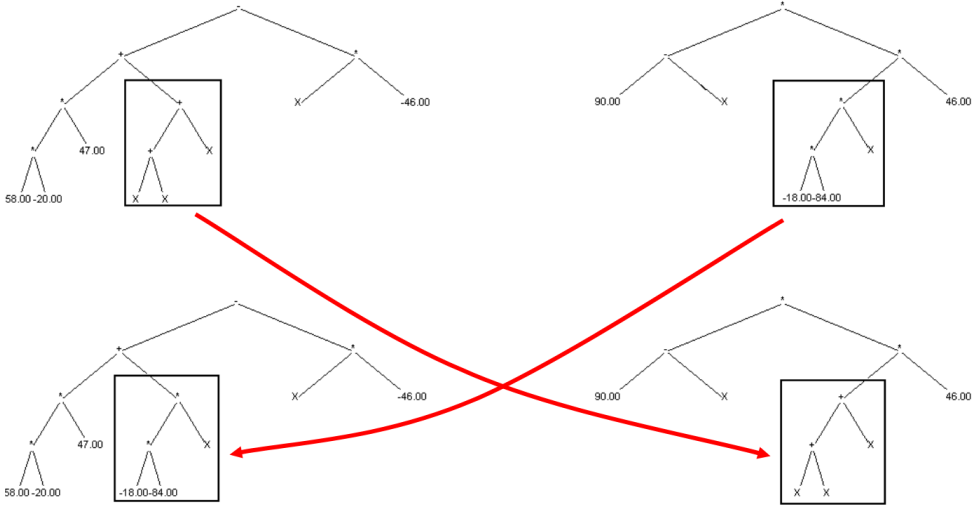
/*4*/rrl<X>#5<,<GX>RR<<llX>LX>LX>>>X

f4 crossing-over idea





f4 crossing-over idea



Mutation and repair

- Mutation: modification of every element of a genotype. Small, local changes
- Validity test: many aspects
- Repair: attempt to correct an invalid genotype
  - ensure each property value is within allowed interval
  - fix neural links
  - contextual: match brackets etc.
  - ...

References I

[HK08] Maciej Hapke and Maciej Komosinski. "Evolutionary Design of Interpretable Fuzzy Controllers". In: *Foundations of Computing and Decision Sciences* 33.4 (2008), pp. 351–367. URL: http://www.framsticks.com/files/common/EvolveInterpretableFuzzyControl.pdf.

[HKW03] Maciej Hapke, Maciej Komosinski, and Dawid Waclawski. "Application of Evolutionarily Optimized Fuzzy Controllers for Virtual Robots". In: *Proceedings of the 7th Joint Conference on Information Sciences*. North Carolina, USA: Association for Intelligent Machinery, Sept. 2003, pp. 1605–1608. URL: http://www.framsticks.com/files/common/EvolvedFuzzyControl_CINC2003.pdf.

[JK06] Jacek Jelonek and Maciej Komosinski. "Biologically-inspired Visual-motor Coordination Model in a Navigation Problem". In: *Knowledge-Based Intelligent Information and Engineering Systems*. Ed. by Bogdan Gabrys, Robert Howlett, and Lakhmi Jain. Vol. 4253. Lecture Notes in Computer Science. Berlin/Heidelberg: Springer, 2006, pp. 341–348. DOI: 10.1007/11893011_44. URL: http://www.framsticks.com/files/common/BiologicallyInspiredVisualMotorCoordinationModel.pdf.

[KR01] Maciej Komosinski and Adam Rotaru-Varga. "Comparison of different genotype encodings for simulated 3D agents". In: *Artificial Life Journal* 7.4 (Fall 2001), pp. 395–418. DOI: 10.1162/106454601317297022. URL: http://www.framsticks.com/files/common/ComparisonGeneticEncodings3DAgents.pdf.

[KU04] Maciej Komosinski and Szymon Ulatowski. "Genetic mappings in artificial genomes". In: *Theory in Biosciences* 123.2 (Sept. 2004), pp. 125–137. DOI: 10.1016/j.thbio.2004.04.002. URL: http://www.framsticks.com/files/common/GeneticMappingsInArtificialGenomes.pdf.