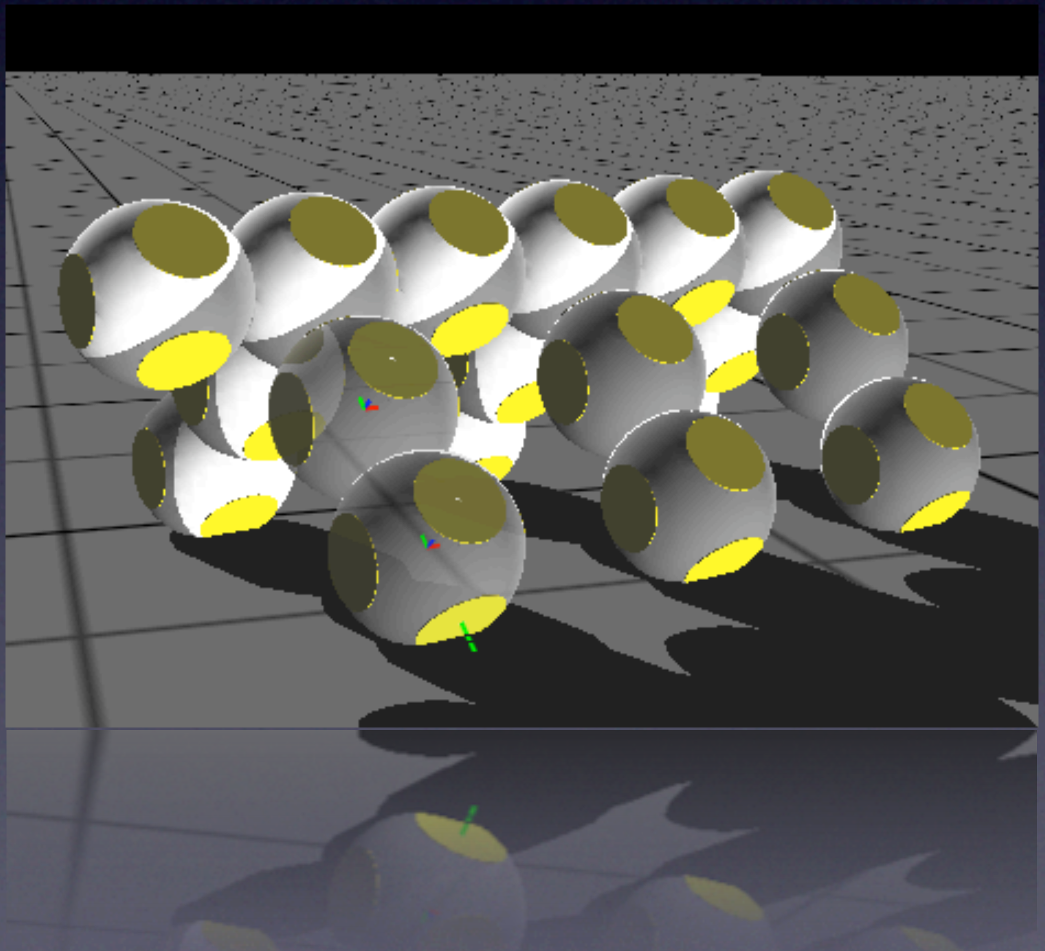
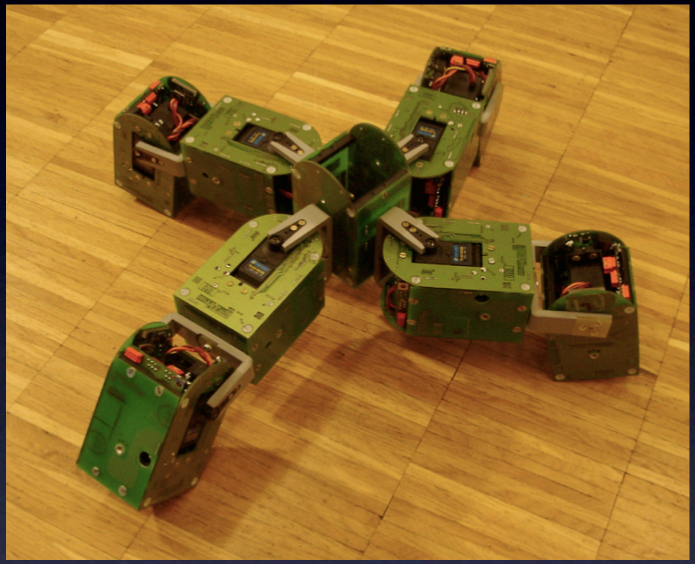
Roombots
CPGs, Symmetries and Online Learning



TULEU Alexandre

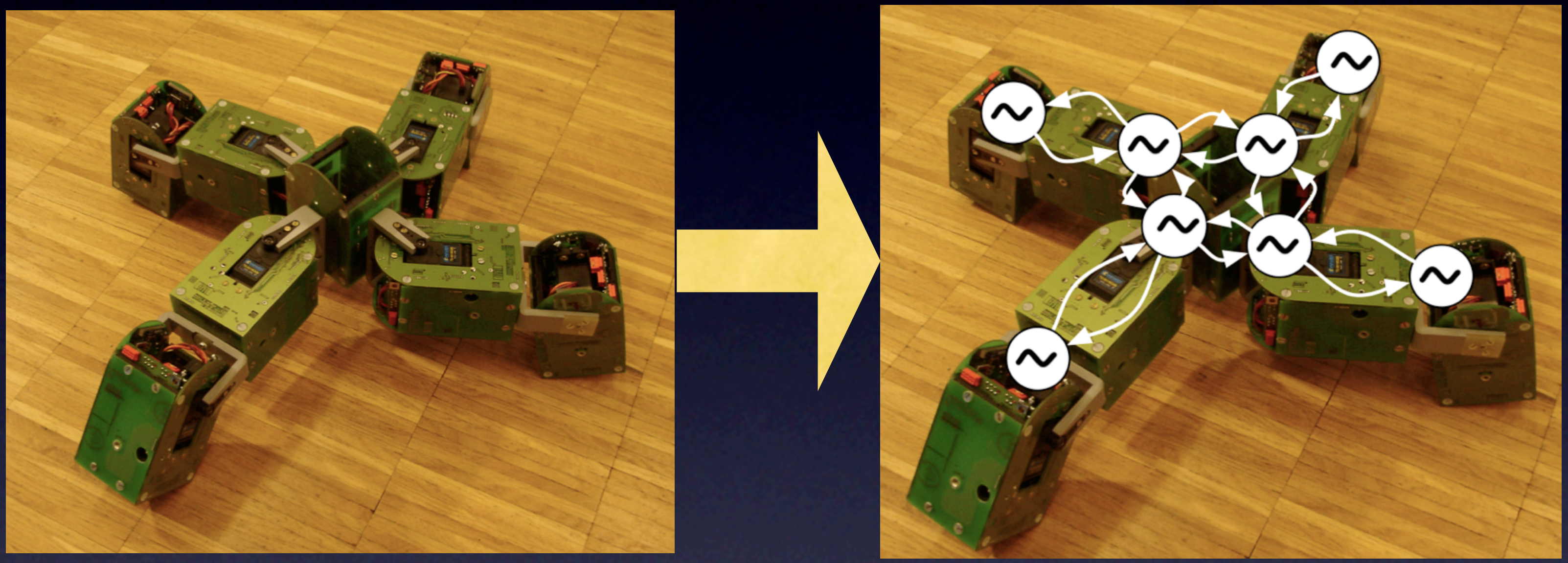
Semester Project

How the problem was formerly adressed.



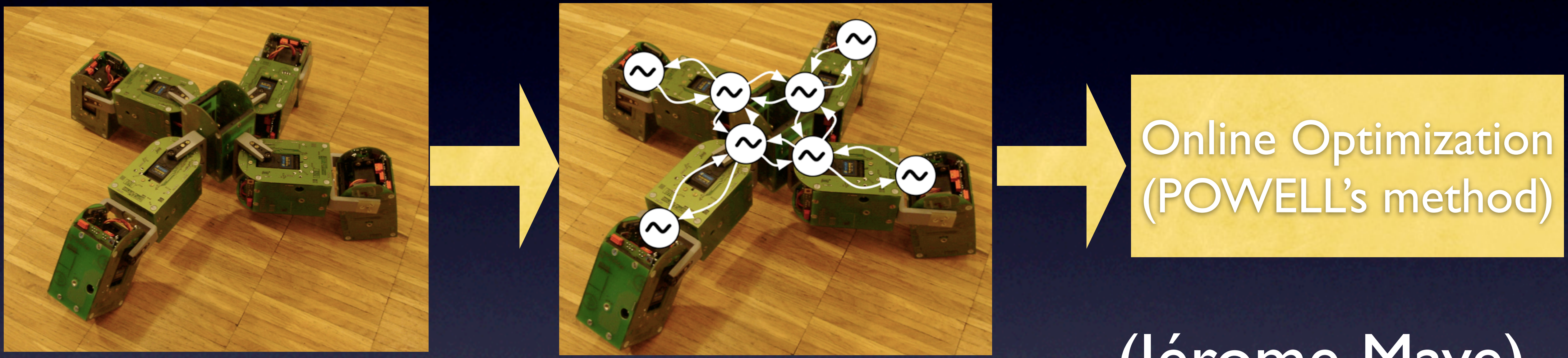
(Jérome Maye)

How the problem was formerly adressed.



(Jérome Maye)

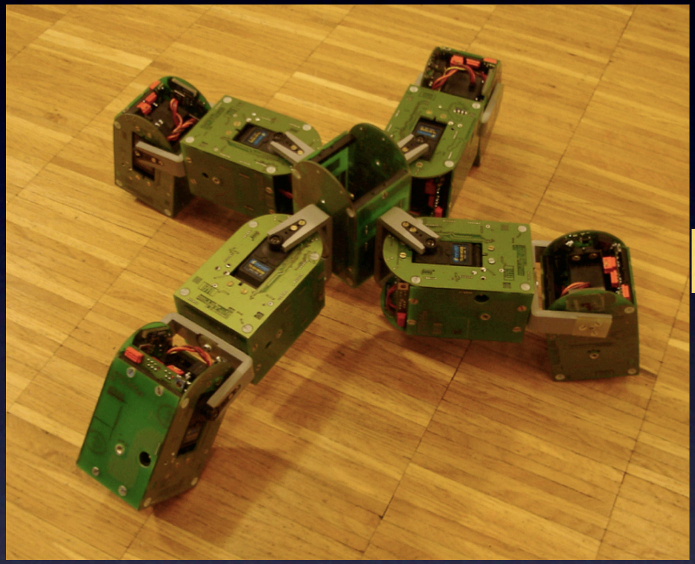
How the problem was formerly adressed.

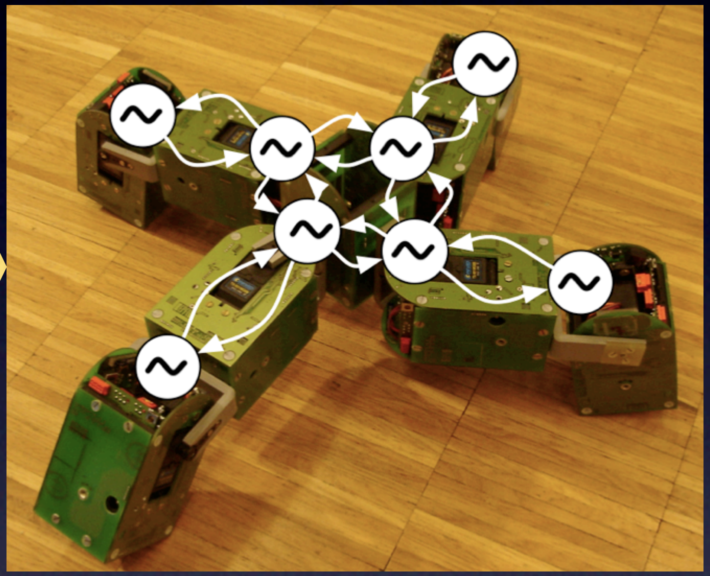


Online Optimization (POWELL's method)

(Jérome Maye)

How the problem was formerly adressed.





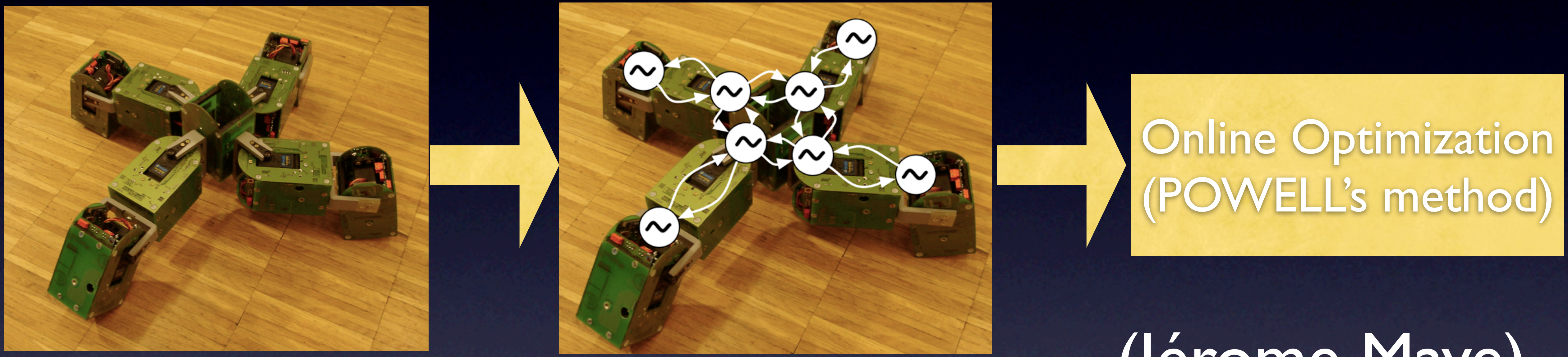
Online Optimization
(POWELL's method)

(Jérome Maye)

Problems encountered :

- Manual design of the CPG is not scalable !

How the problem was formerly adressed.



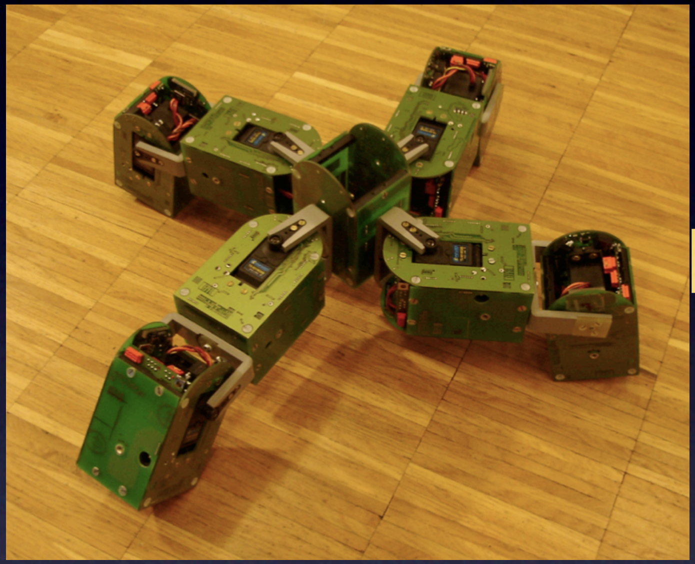
Online Optimization (POWELL's method)

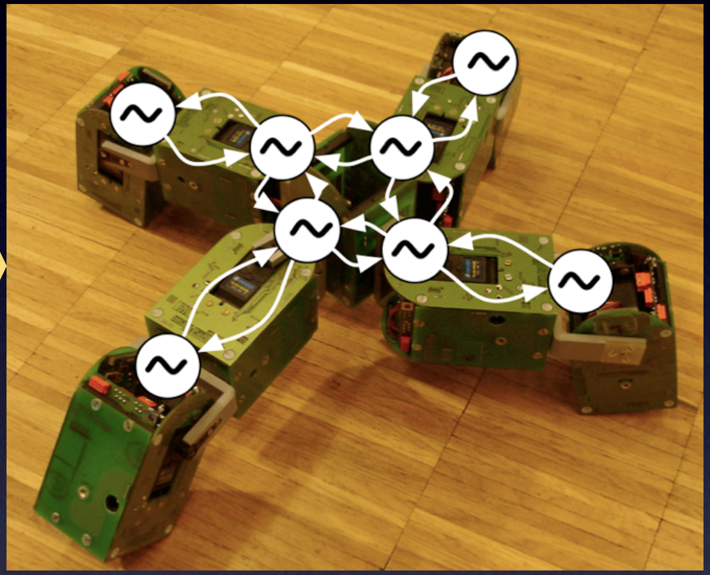
(Jérome Maye)

Problems encountered :

- Manual design of the CPG is not scalable !
- Optimized search for the topology of the network neither !

How the problem was formerly adressed.





Online Optimization (POWELL's method)

(Jérome Maye)

Problems encountered :

- Manual design of the CPG is not scalable !
- Optimized search for the topology of the network neither !
- Powell's method less suited for Roombots (multi-modal objective function)

Goals of the project.

Goals of the project.

- Continue the previous work, and test new CPG models and optimization algorithms.

Goals of the project.

- Continue the previous work, and test new CPG models and optimization algorithms.
- Find a way to automatically design CPG networks.

Goals of the project.

- Continue the previous work, and test new CPG models and optimization algorithms.
- Find a way to automatically design CPG networks.

Goals of the project.

- Continue the previous work, and test new CPG models and optimization algorithms.
- Find a way to automatically design CPG networks.

  - Presentation of some software tools created.

Goals of the project.

- Continue the previous work, and test new CPG models and optimization algorithms.
- Find a way to automatically design CPG networks.

    - Presentation of some software tools created.
    - Study of a special case : the quadruped Roombots

Goals of the project.

- Continue the previous work, and test new CPG models and optimization algorithms.
- Find a way to automatically design CPG networks.

  - Presentation of some software tools created.
  - Study of a special case : the quadruped Roombots
  - Presentation of an automatic processus for designing CPG.

Making tools for Roombots

Making tools for Roombots

- libCPG : a library for manipulating CPG.

Making tools for Roombots

- libCPG : a library for manipulating CPG.
- worldMaker

Making tools for Roombots

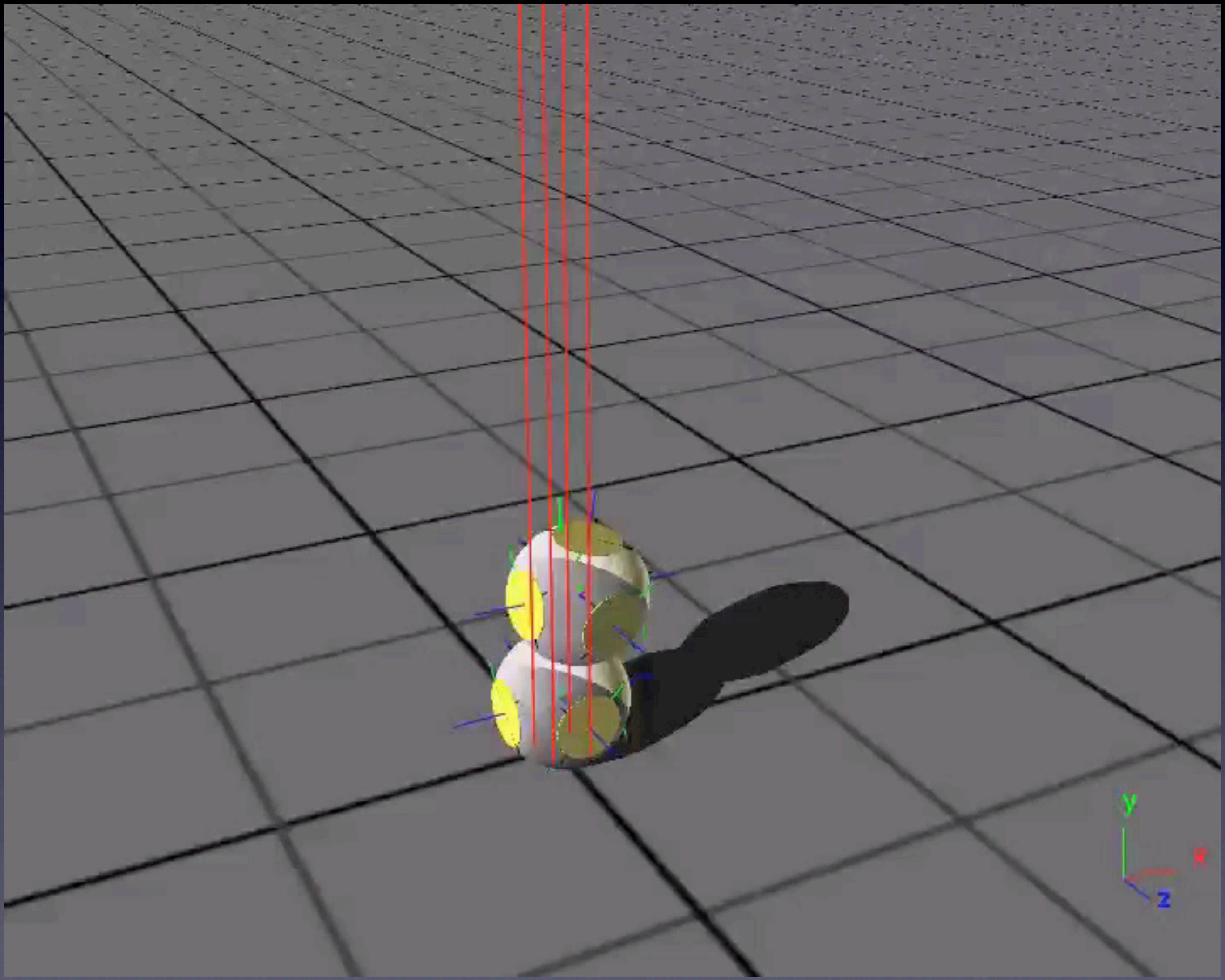
- libCPG : a library for manipulating CPG.
- worldMaker
- Enhancing the collision detection for Roombots.

Making tools for Roombots

- libCPG : a library for manipulating CPG.
- worldMaker
- Enhancing the collision detection for Roombots.
  - The modules were previously modelised by two spheres.

Results

Results



Study of a special case : the quadruped Roombots.

Study of a special case : the quadruped Roombots.

- Goal : get experienced with modular robotic locomotion.

Study of a special case : the quadruped Roombots.

- Goal : get experienced with modular robotic locomotion.
    - Analyse the choices we made, in order to synthesize them in an automatic process .

Symmetries and locomotion

Symmetries and locomotion

- Symmetries are a generality in animals. The most efficient morphologies for modular robots have a symmetry. (Daniel Marbach)

Symmetries and locomotion

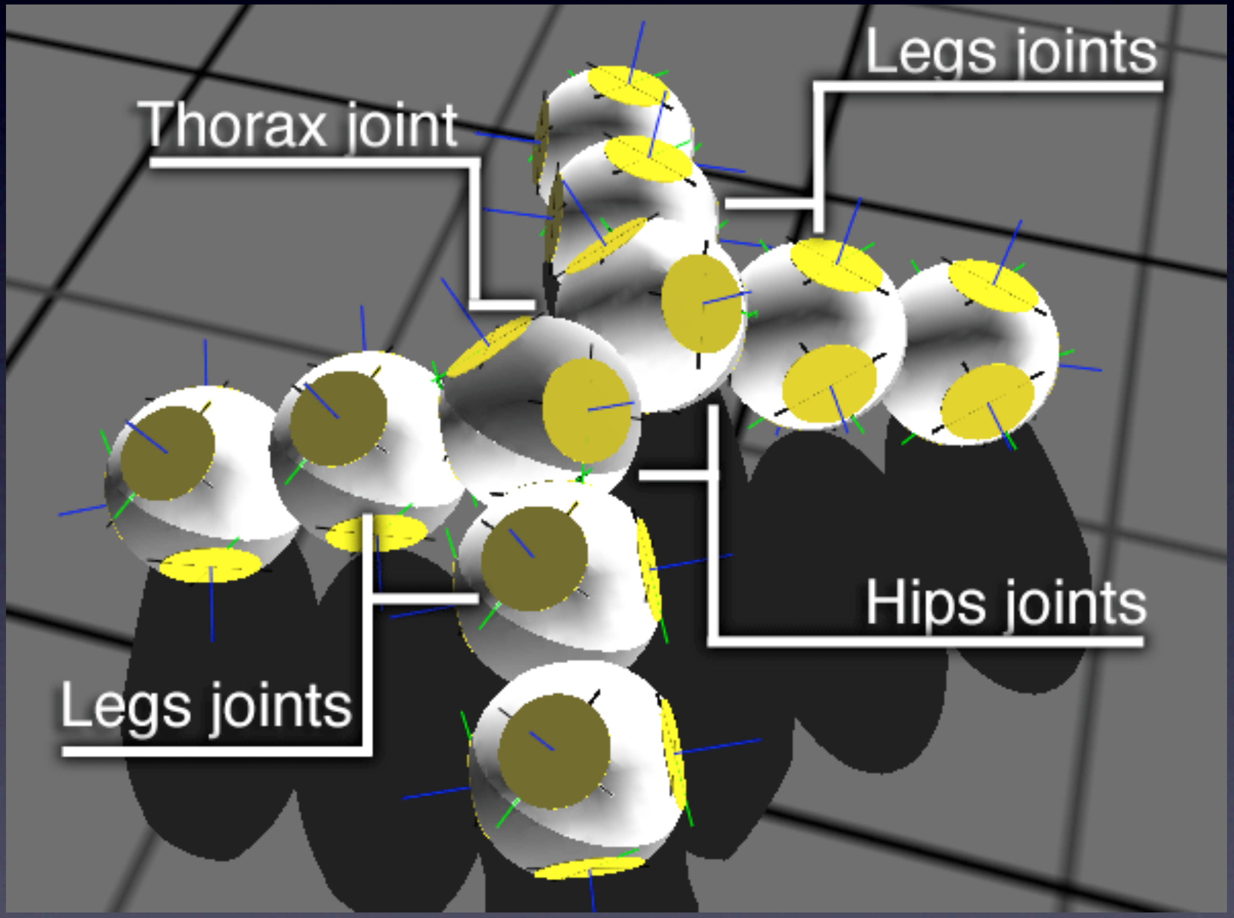
- Symmetries are a generality in animals. The most efficient morphologies for modular robots have a symmetry. (Daniel Marbach)
- Symmetries in CPG help to determine the resulting pattern (Ludovic Righetti, Golubitsky).

Symmetries and locomotion

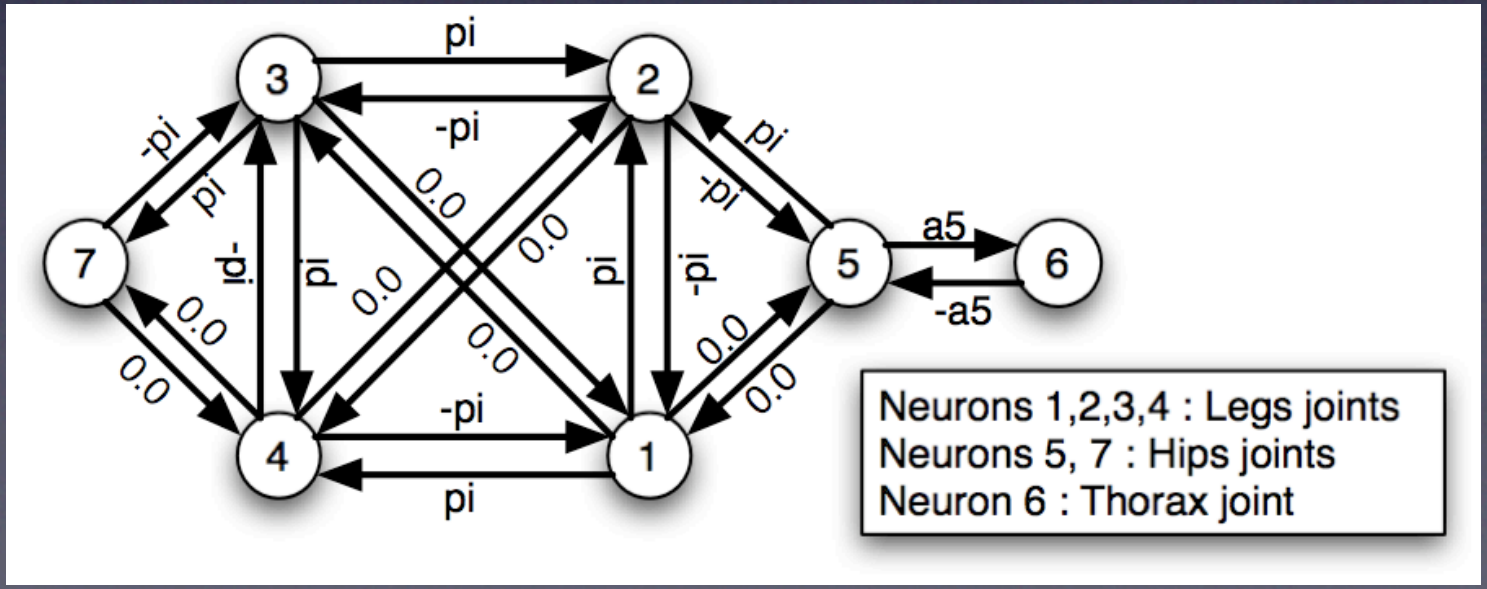
- Symmetries are a generality in animals. The most efficient morphologies for modular robots have a symmetry. (Daniel Marbach)
- Symmetries in CPG help to determine the resulting pattern (Ludovic Righetti, Golubitsky).
- Then it helps us to reduce our search space, by adressing the same value to symmetric peers. (Sandra Wieser)

Design of the robot.

- Try to find the most symmetric structure.
- Make the legs to have the same motion.
- Make the structure able to assymetrize itself, in order to select which leg to use (action on the Thorax joint).



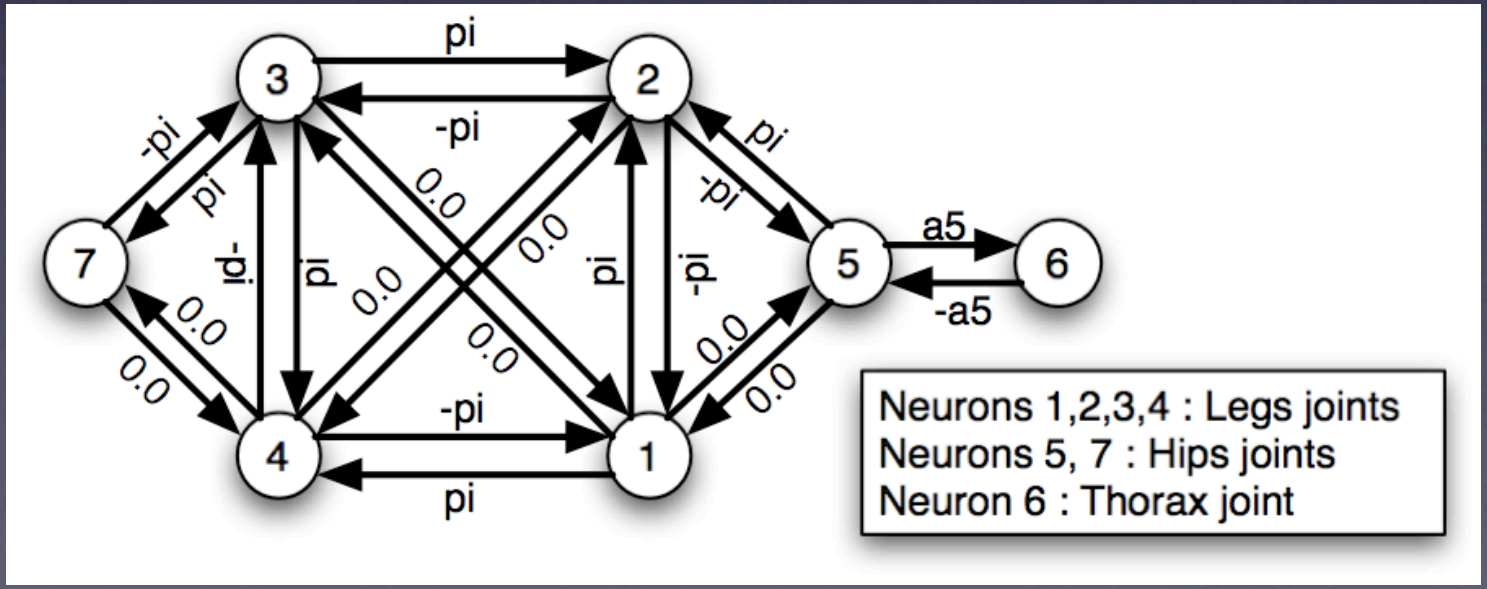
Design of the CPG



Neurons 1,2,3,4 : Legs joints
Neurons 5, 7 : Hips joints
Neuron 6 : Thorax joint

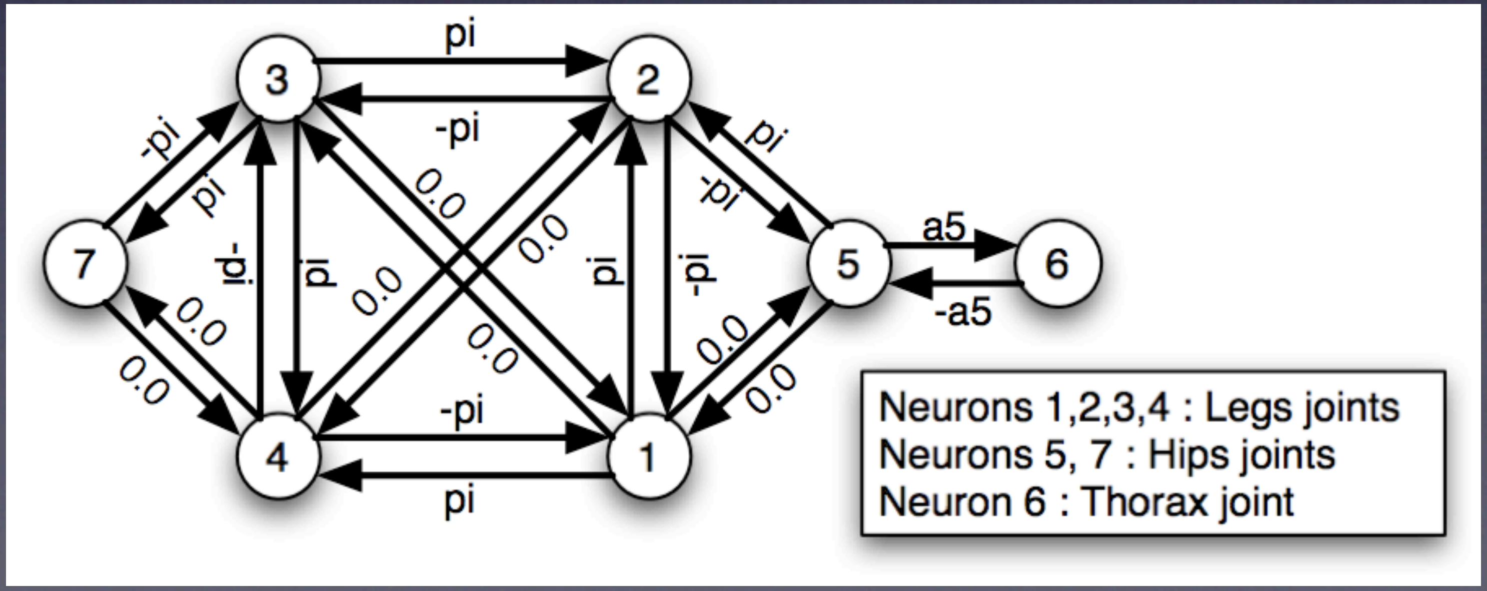
Design of the CPG

- Designed to provide a symmetrical trot.



Design of the CPG

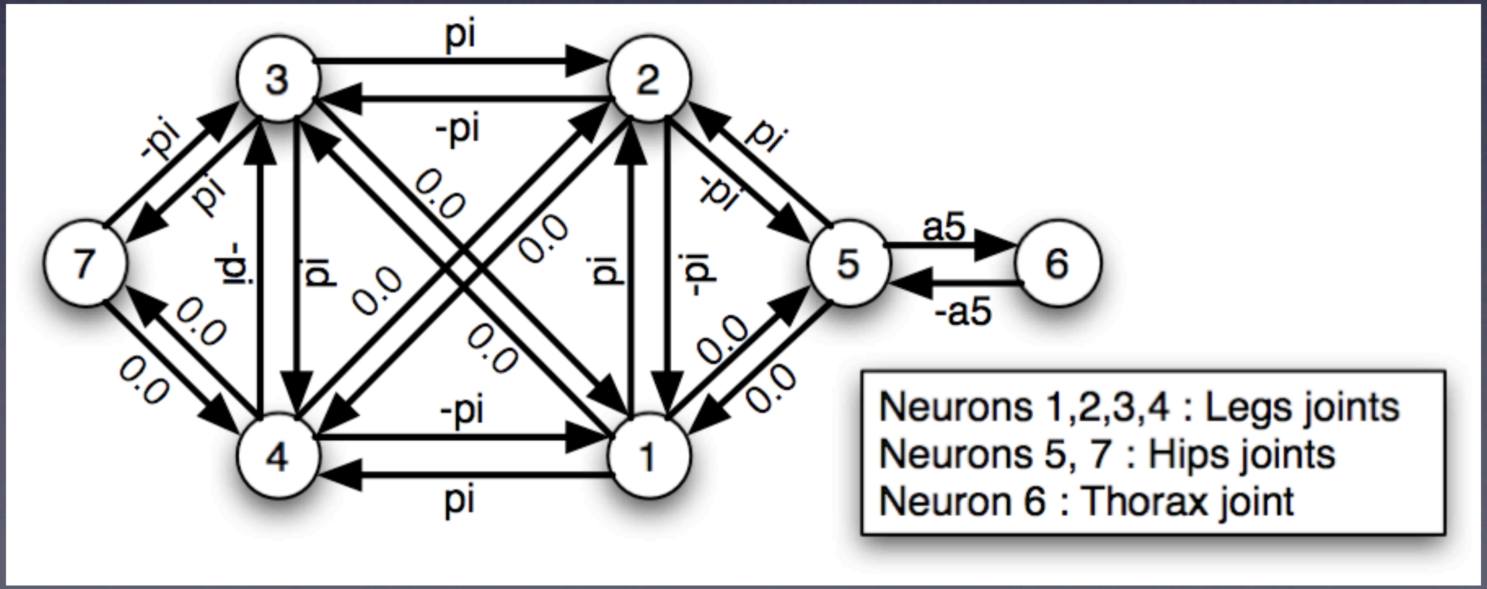
- Designed to provide a symmetrical trot.
- Graft the central part of the neuron to synchronize with the leg part.



Neurons 1,2,3,4 : Legs joints
Neurons 5, 7 : Hips joints
Neuron 6 : Thorax joint

Design of the CPG

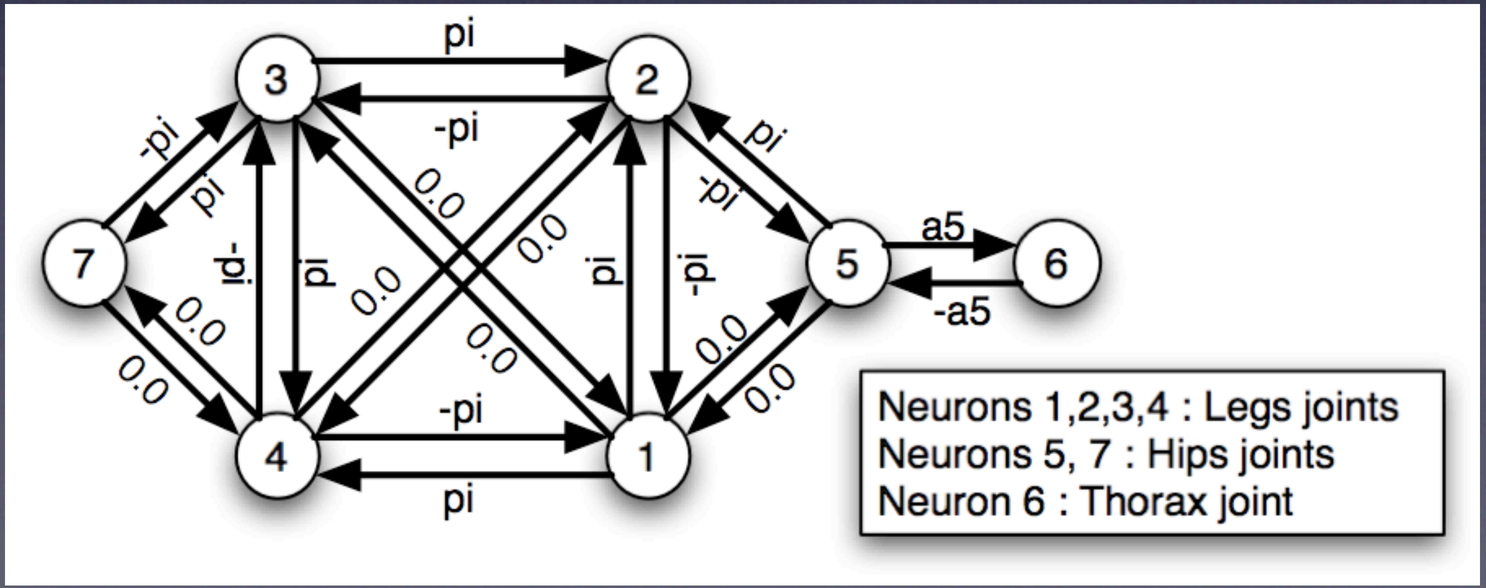
- Designed to provide a symmetrical trot.
- Graft the central part of the neuron to synchronize with the leg part.
- The central part offset is fixed to work around the starting posture, and the synchronization is opened.



Neurons 1,2,3,4 : Legs joints
Neurons 5, 7 : Hips joints
Neuron 6 : Thorax joint

Design of the CPG

- Designed to provide a symmetrical trot.
- Graft the central part of the neuron to synchronize with the leg part.
- The central part offset is fixed to work around the starting posture, and the synchronization is opened.
- The leg parts of the structure have a fixed synchronization, but a free offset.

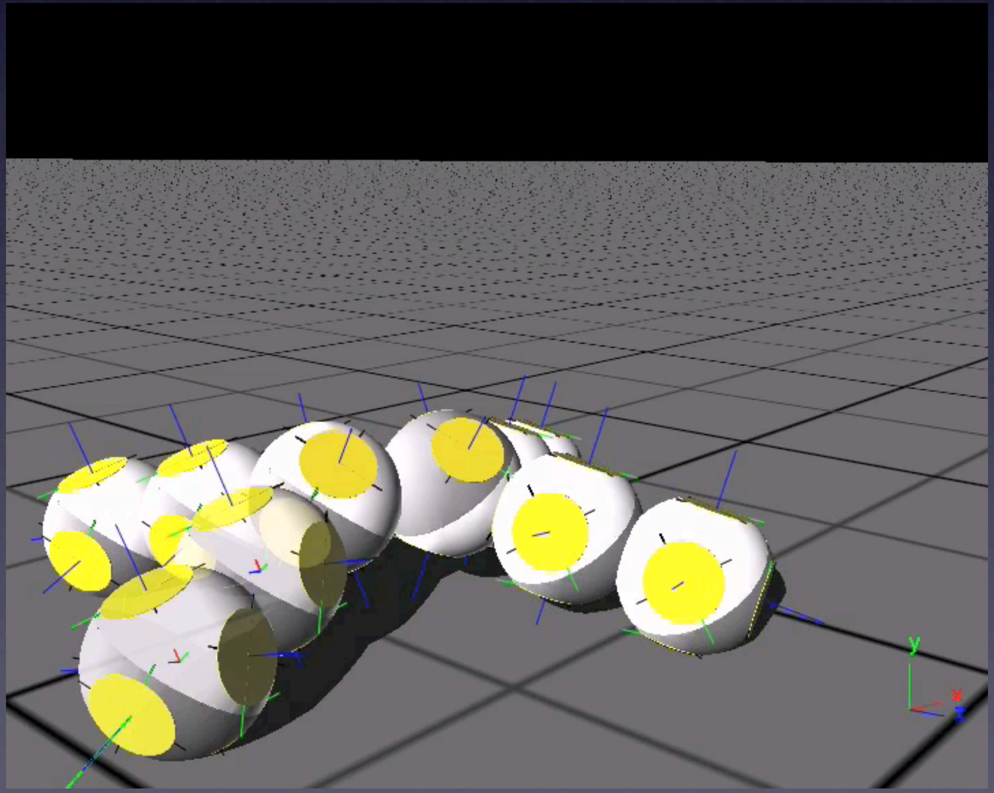


Neurons 1,2,3,4 : Legs joints
Neurons 5, 7 : Hips joints
Neuron 6 : Thorax joint

Results obtained.

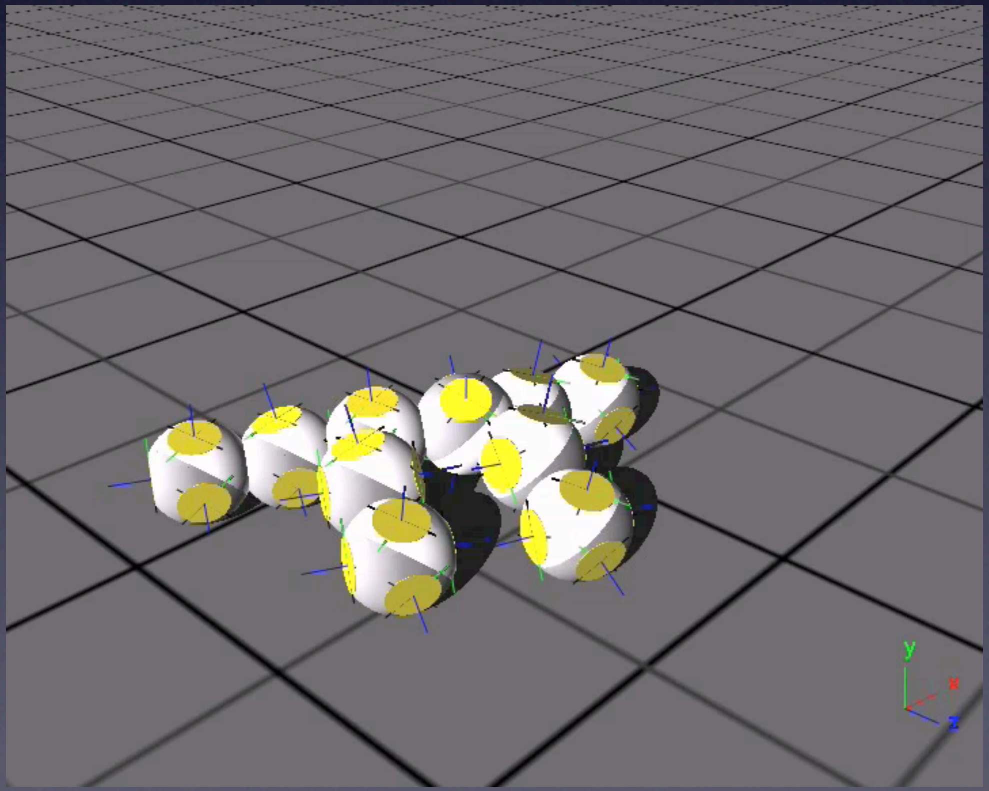
With fixed frequency

With optimized frequency

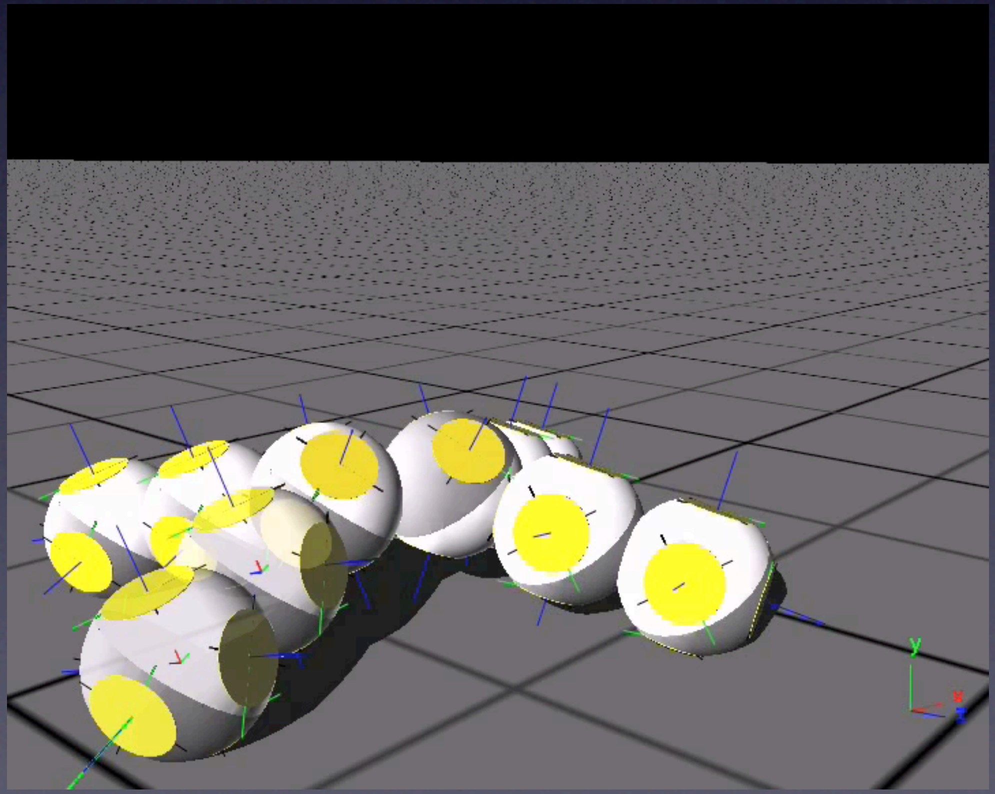


Results obtained.

With fixed frequency

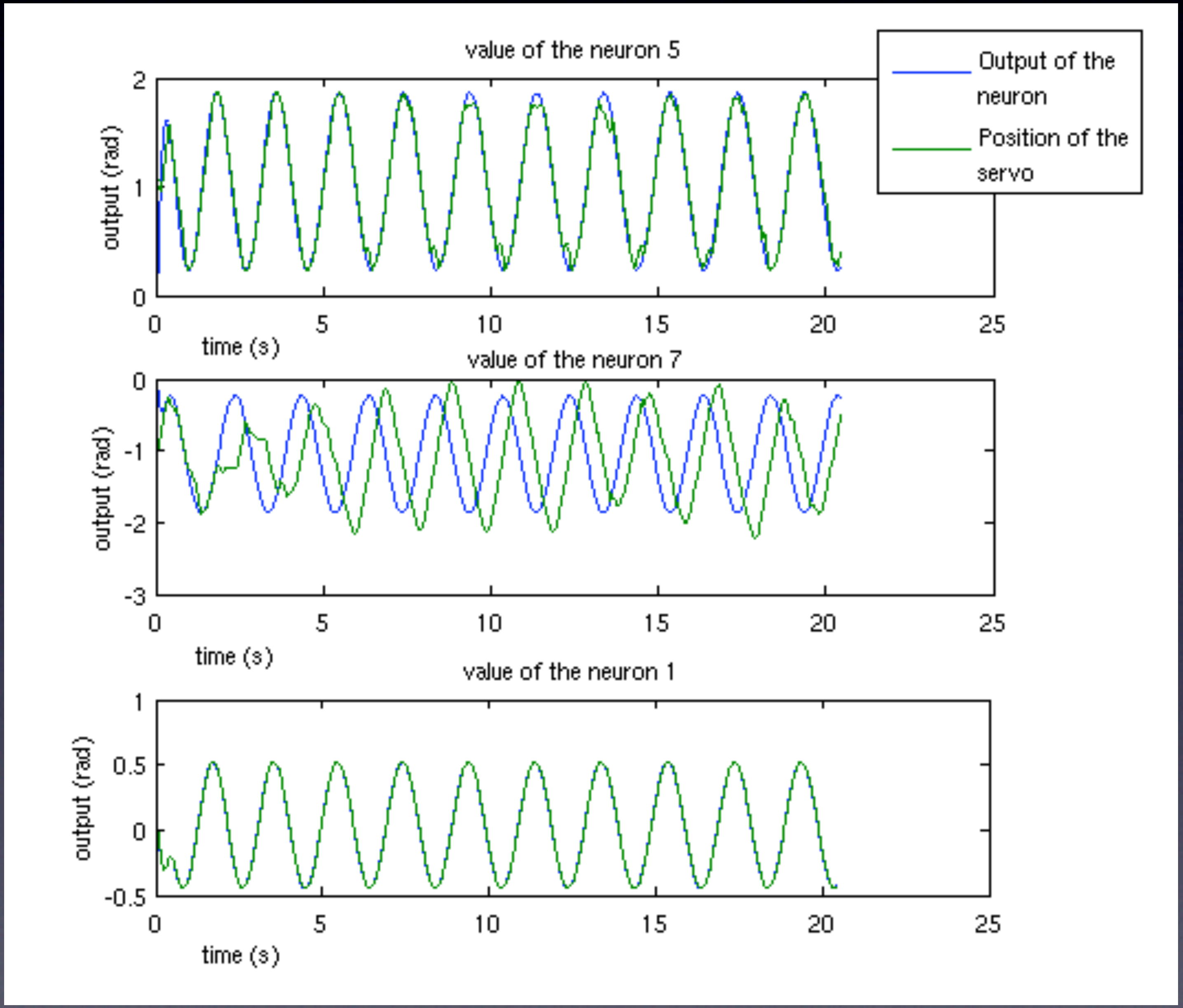


With optimized frequency



Problems : Servos limitation

- Have a limited (slow) speed : 25 tr.min^{-1}
- Should add some constraints on the search space.
- Should add some "control feedback" in the CPG.



Towards an automatic designer for CPG

Choices we made for the quadruped :

Towards an automatic designer for CPG

Choices we made for the quadruped :

- Search for the most symmetrical robot.

Towards an automatic designer for CPG

Choices we made for the quadruped :

- Search for the most symmetrical robot.
- Separate “leg” and “body” parts.

Towards an automatic designer for CPG

Choices we made for the quadruped :

- Search for the most symmetrical robot.
- Separate “leg” and “body” parts.
- Set the central point of the body part neuron, in order to “play” with the symmetry of the structure.

Towards an automatic designer for CPG

Choices we made for the quadruped :

- Search for the most symmetrical robot.
- Separate “leg” and “body” parts.
- Set the central point of the body part neuron, in order to “play” with the symmetry of the structure.
- Avoid to add neurons that command useless joints.

Towards an automatic designer for CPG

Choices we made for the quadruped :

- Search for the most symmetrical robot.
- Separate "leg" and "body" parts.
- Set the central point of the body part neuron, in order to "play" with the symmetry of the structure.
- Avoid to add neurons that command useless joints.

Steps of our algorithm :

Towards an automatic designer for CPG

Choices we made for the quadruped :

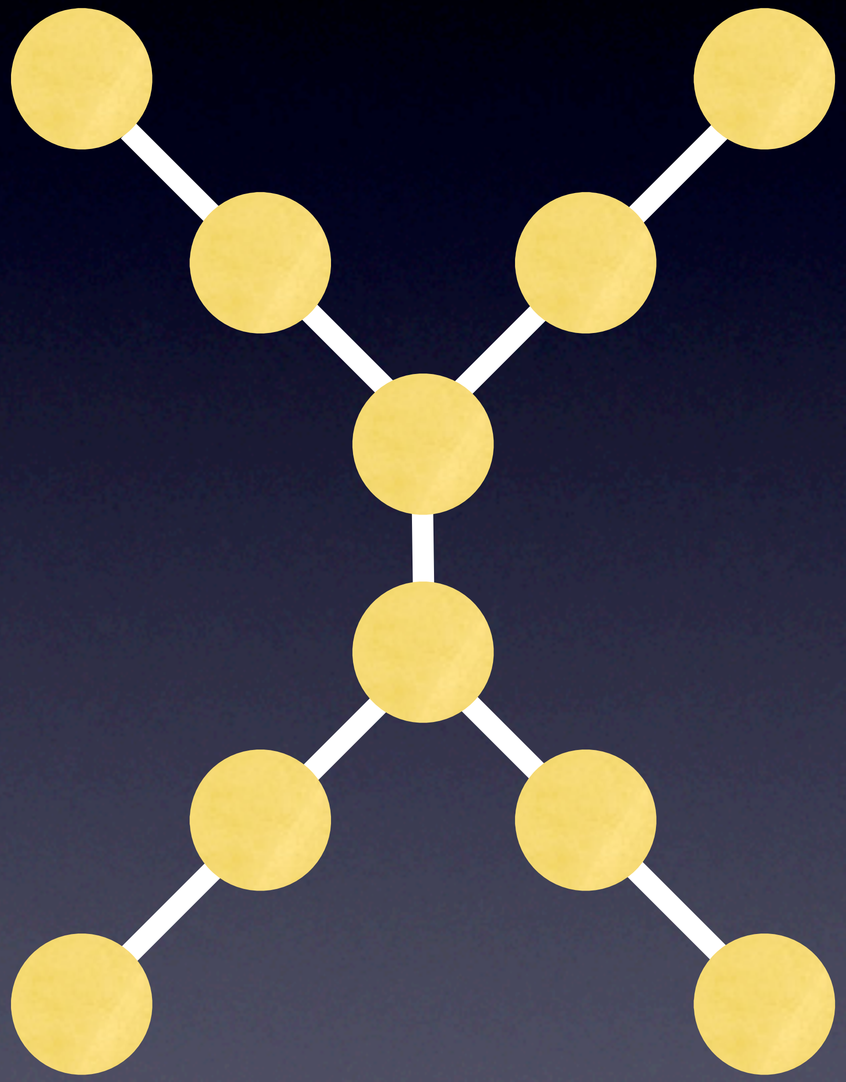
- Search for the most symmetrical robot.
- Separate “leg” and “body” parts.
- Set the central point of the body part neuron, in order to “play” with the symmetry of the structure.
- Avoid to add neurons that command useless joints.

Steps of our algorithm :

- Classify the “leg” and “body” parts.
- Find the symmetries in the structure
- Set up the CPG, according to the symmetries, classification and heuristic rules.

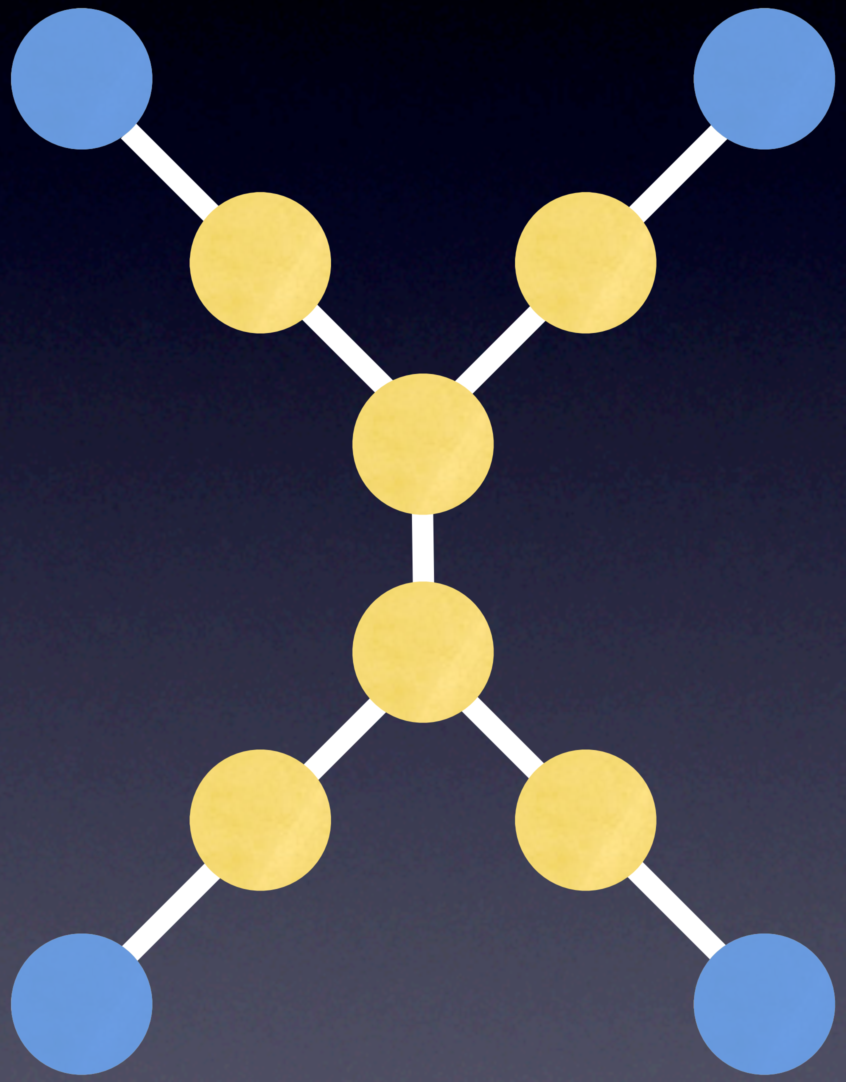
Module Classification

- We forbid Cycles. The Graph is a tree.
- We classify all the leaves as “legs” part.
- “Legs” property is propaged to parent, if it has only one child.



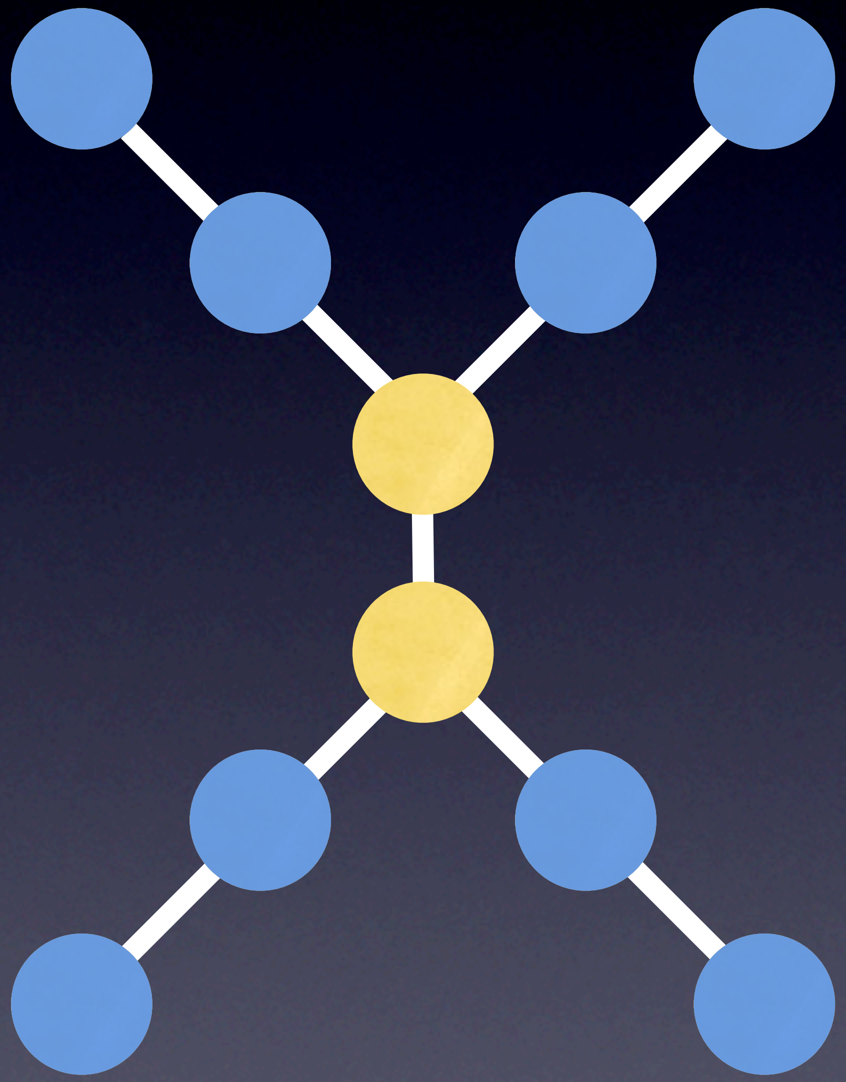
Module Classification

- We forbid Cycles. The Graph is a tree.
- We classify all the leaves as “legs” part.
- “Legs” property is propaged to parent, if it has only one child.



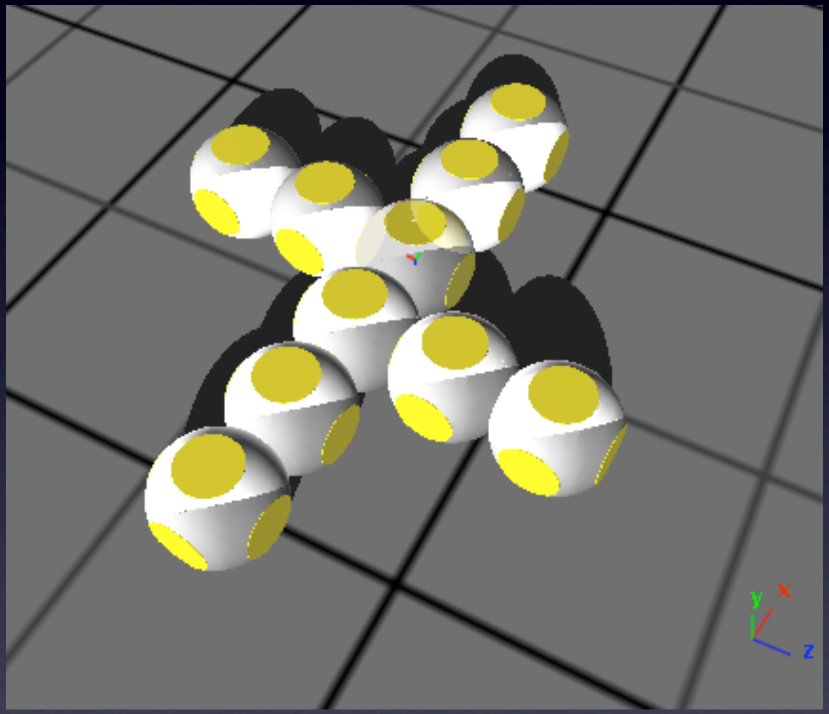
Module Classification

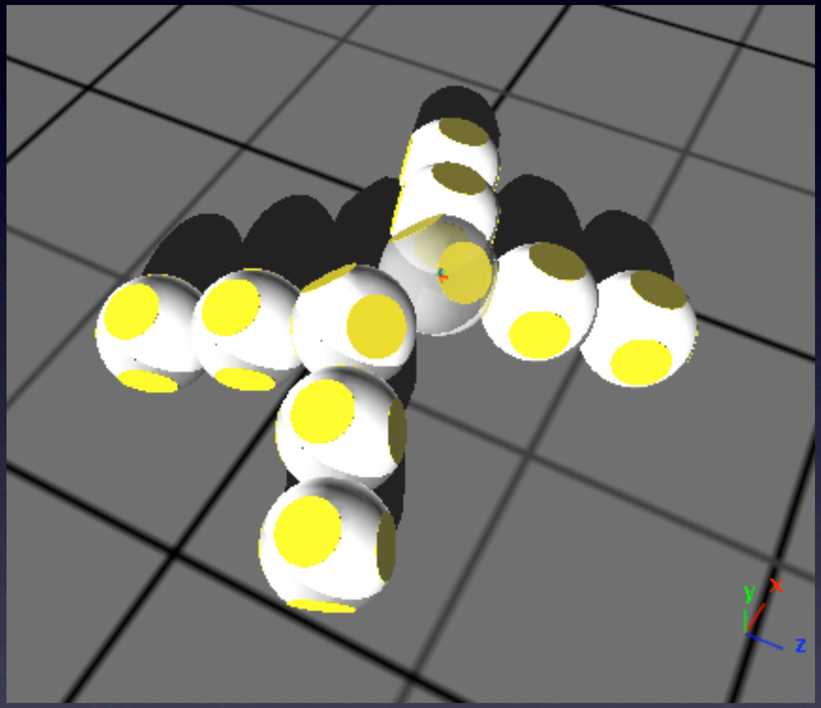
- We forbid Cycles. The Graph is a tree.
- We classify all the leaves as “legs” part.
- “Legs” property is propaged to parent, if it has only one child.



Find the symmetries

Hierarchical vs Morphological





Hierarchical symmetries is a prerequisite for Morphological ones.

Morphological symmetries can be found as the result of an **Optimization Process** on the servo position.

Estimating how much Morphologically symmetric a structure is.

- For a given position of the servo, we have to check for all the hierarchical symmetries, how much they are close to a spatial symmetry between modules.
- As a result of Graph Theory, this problem can be reduced to test symmetries among several pairs of modules.
- Thanks to the Roombots kinematics model provided by (Mikaël Mayer), the expression of how much two modules are symmetric can be adressed.

Therefore, we can provide an **objective function** for our optimization algorithm.

Estimating how much two modules are symmetric.

- Thanks to the given servo position and the kinematics models, we can compute the Rotational Matrix R from one module to another .
- The two modules are symmetric when : R^2=Id. (axial symmetry)
- “How much they are symmetric” is then the “distance” from R^2 to Id.
- Open questions : Which space to compute the distance in; reflexive symmetries ?

Questions ?