University of Pisa

Master of Science in Computer Science

**Course of Robotics (ROB)**

A.Y. 2017/18

# Robot Architectures

Cecilia Laschi

The BioRobotics Institute

Scuola Superiore Sant'Anna, Pisa

cecilia.laschi@santannapisa.it

http://didawiki.cli.di.unipi.it/doku.php/magistraleinformatica/rob/start

THE BIOROBOTICS INSTITUTE

Scuola Superiore Sant'Anna

# Robot definition

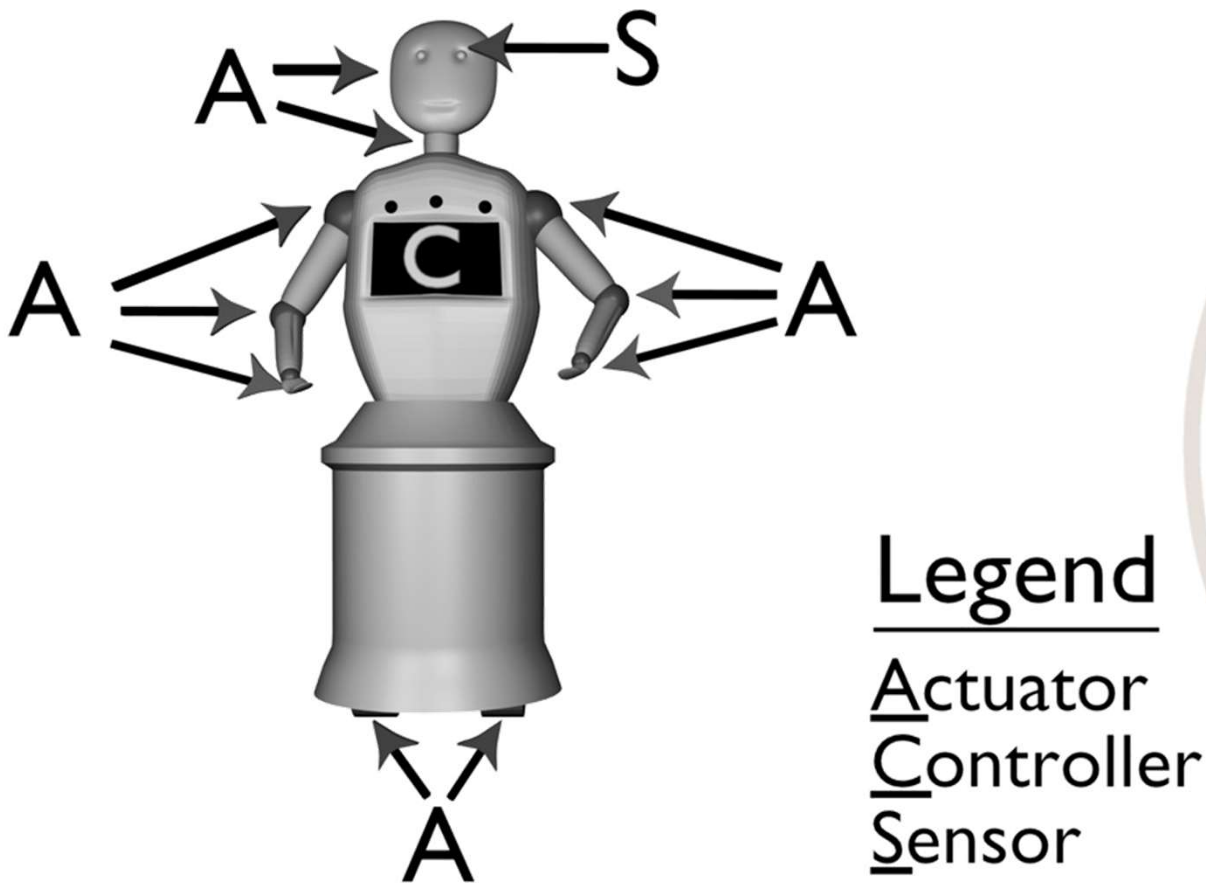
A *robot* is an autonomous system
which exists in the physical world,
can sense its environment,
and can act on it to achieve some goals

Maja J Mataric, *The Robotics Primer*, The MIT Press, 2007

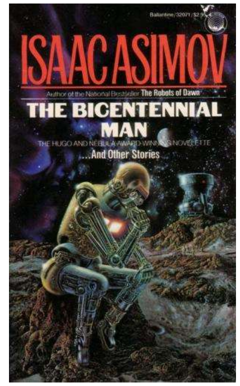
# What's in a robot?
# Robot components

THE BIOROBOTICS INSTITUTE

Scuola Superiore Sant'Anna




Maja J Mataric, *The Robotics Primer*, The MIT Press, 2007





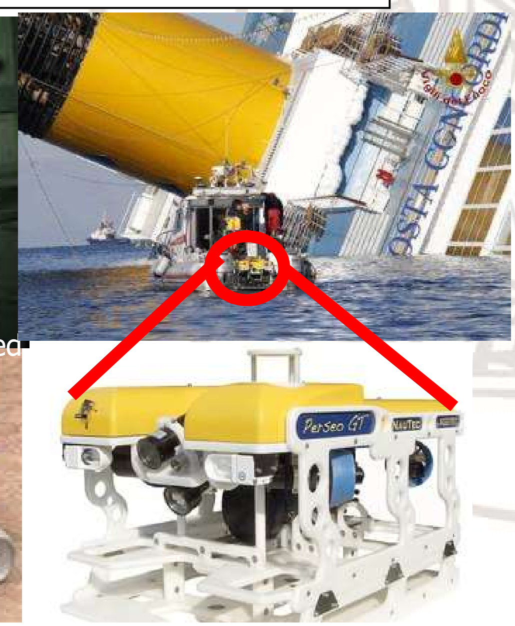
**Industrial robotics:**
birth and growth of theories and techniques for robot control

**Service robotics:**
birth and growth of theories and techniques for robot **behaviour** (perception & action) control

Structured environment

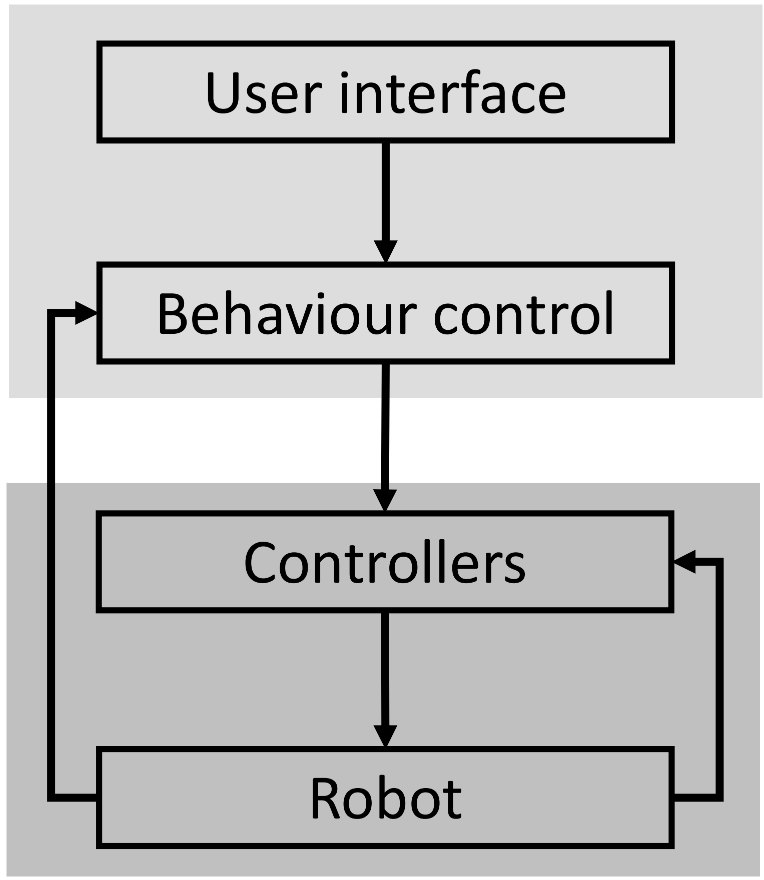
Unstructured environment


Photo: Center for Robot-Assisted Search and Rescue



# Robot behaviour

- High level
- Low level



## The robotic paradigms

- *"A paradigm is a philosophy or set of assumptions and/or techniques which characterize an approach to a class of problems"*
- No one paradigm is right; rather, some problems seem better suited for different approaches.
- Applying the right paradigm makes problem solving easier.

# The robotic paradigms

- Traditionally, there are 3 main paradigms for facing the problem of controlling robot behaviour:
  - Hierarchial paradigm
  - Reactive paradigm
  - Hybrid paradigm

# The robotic paradigms

- The 3 paradigms differ in the way the commonly accepted primitives of robotics are organized
- the commonly accepted primitives of robotics are:
  - **SENSE**: takes information from the robot sensors and produces an output for other functions
  - **PLAN**: takes information from the SENSE or from a world model and produces tasks for the robot
  - **ACT**: takes the tasks for PLAN and produces output commands for the robot actuators

# The robotic paradigms

The 3 paradigms can be described in 2 ways:

- By the relationships between the 3 commonly accepted primitives of robotics
- By the way sensory data is processed and distributed thorugh the system

# The robotic paradigms

## Information flow

| ROBOT PRIMITIVES | INPUT | OUTPUT |
|---|---|---|
| **SENSE** | Sensor data | Sensed information |
| **PLAN** | Information (sensed and/or cognitive) | Directives |
| **ACT** | Sensed information or directives | Actuator commands |

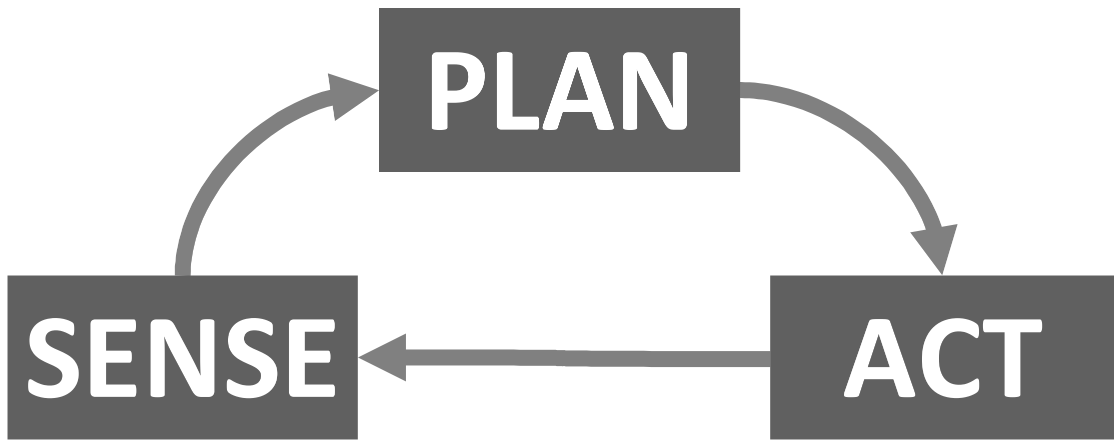
# Robotic architectures

- Provide a principled way of organizing a control system
- Impose constraints on the way the control problem can be solved
- Describe a set of architectural components and how they interact

  -> building blocks of programming a robot
- Criteria for evaluating an architecture:
  - Modularity
  - Niche targettability
  - Portability
  - Robustness

# Robot behaviour

Primitive functions




Hierarchical architectures

# Hierarchical architectures

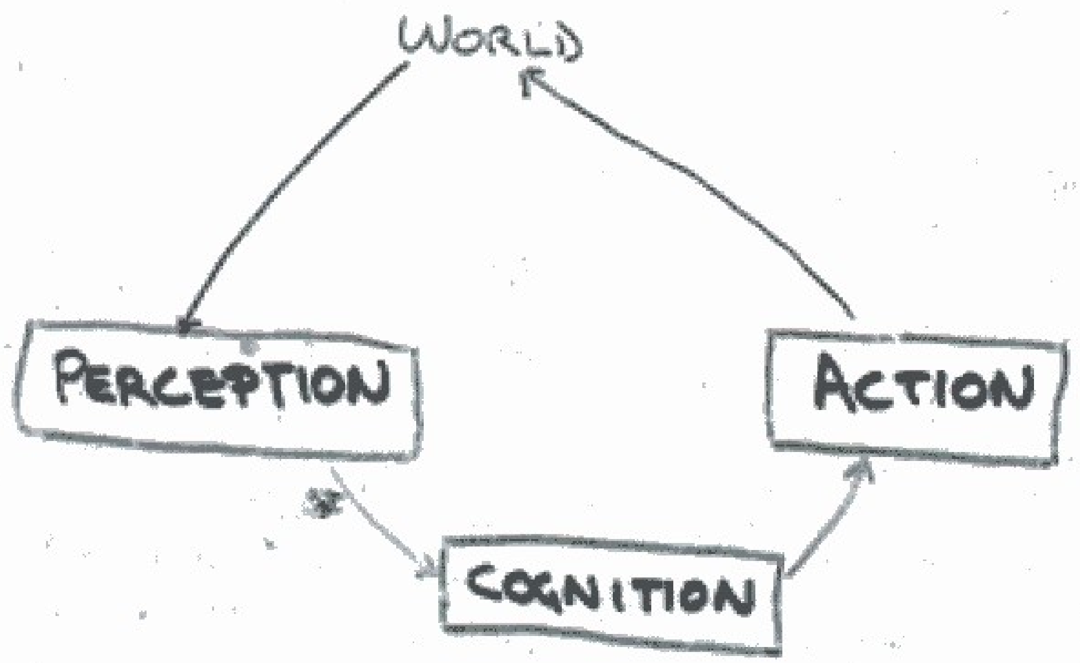
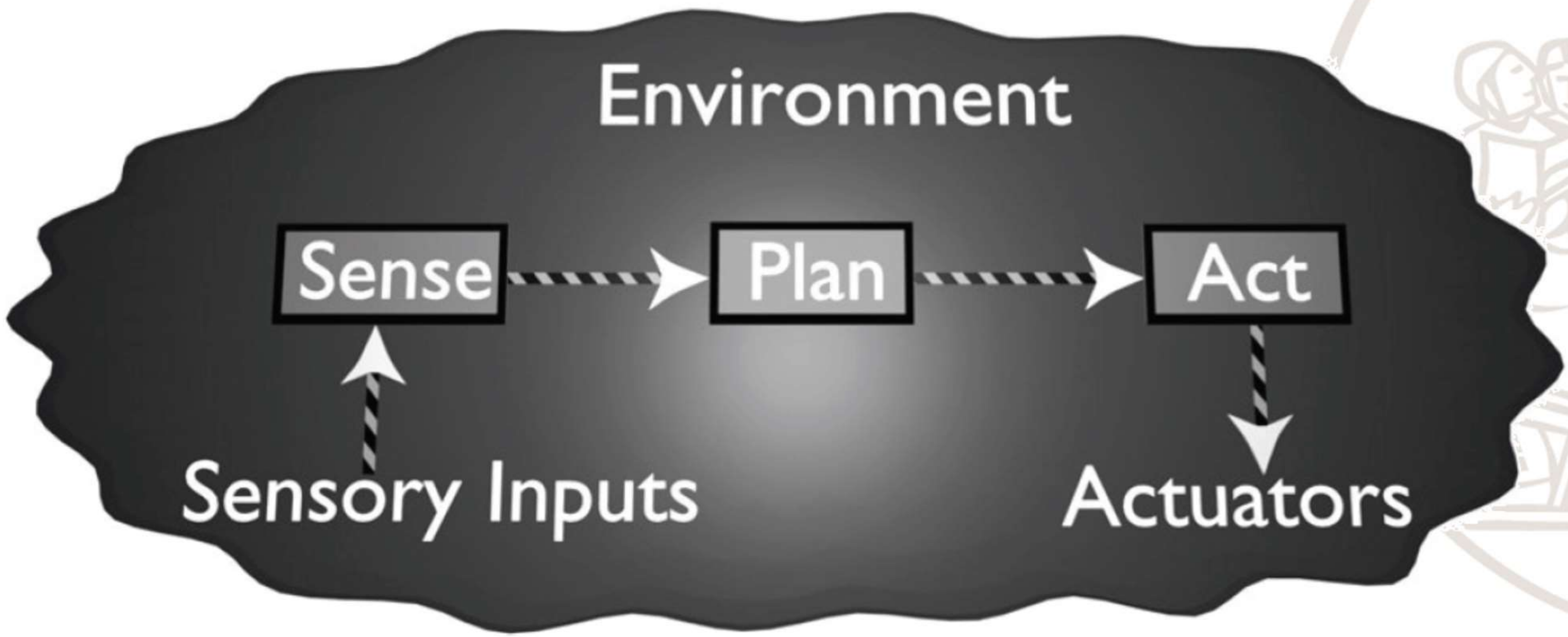



**Figure 1:** The traditional model where cognition mediates between perceptions and plans of actions.

*Tratto da R. Brooks, Cambrian Intelligence*, MIT Press, 1999

# Hierarchical architectures



# Hierarchical paradigm

| ROBOT PRIMITIVES | INPUT | OUTPUT |
|---|---|---|
| SENSE | Sensor data | Sensed information |
| PLAN | Information (sensed and/or cognitive) | Directives |
| ACT | Sensed information or directives | Actuator commands |

# Hierarchical architectures

- Cognition is used to interpret perception and for planning robot tasks

- The SENSE primitive generates a world description, used by the PLAN, which produces a sequence of tasks for the ACT

# Hierarchical architectures

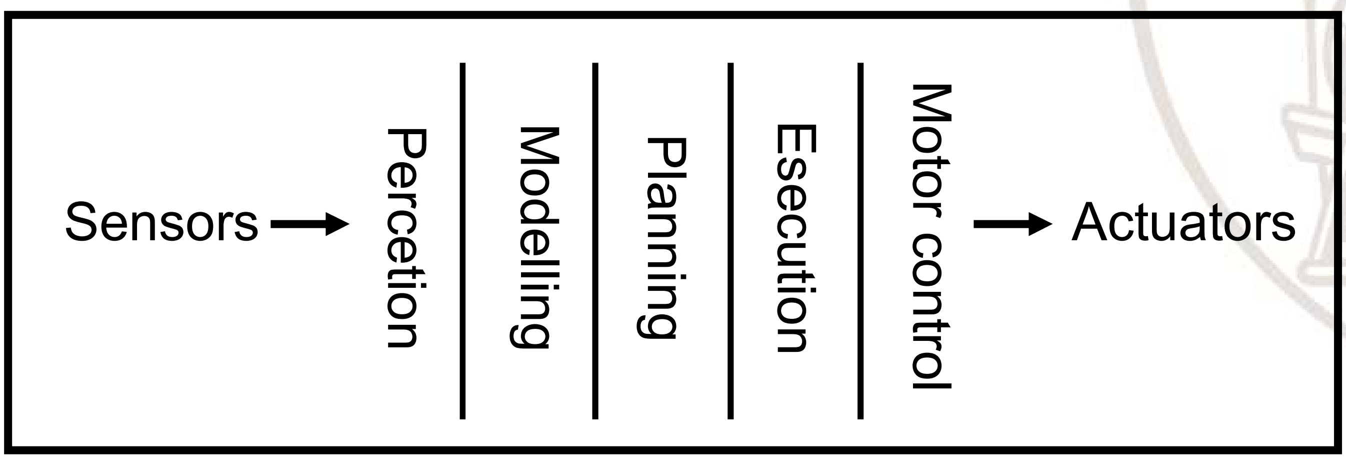
- Perception is used for establishing and maintaining a correspondence between the **internal world model** and the external world.

- Typically, the world model contains:
  - a priori representation of the environment where the robot operates
  - perceived sensory information
  - more information needed for task execution

- The world representation is modified each time the robot perceives the environment and the action plan is established on the basis of such representation

# Hierarchical architectures

- Logical and functional division and distribution of tasks
- Horizontal and sequential decomposition of the chain of the information processed by the central system



# Hierarchical architectures

- Generally, the PLAN primitive is structured in 3 levels:
  - Strategic
  - Tactical
  - Executive
- The highest, or **strategic**, level generates a strategy on the basis of the task to accomplish
- The intermediate, or **tactical**, level generates the commands by interpreting instructions coming from the higher level, or strategic level
- The lowest level, or **executive** level, receives macro-commands generated by the intermediate level and takes care of real-time control of actuators

THE BIOROBOTICS INSTITUTE

Scuola Superiore Sant'Anna

# Hierarchical architectures

## 3-level PLAN structure

**Strategic**
This level generates goals and strategies for achieving those goals which result in the achievement of an overall objective

**What the robot has to do**

↓ ↑

**Tactical**
This level generates the tactis by which a goal is to be achieved

**How to accomplish tasks**

↓ ↑

**Executive**
This level execute the tactical plan

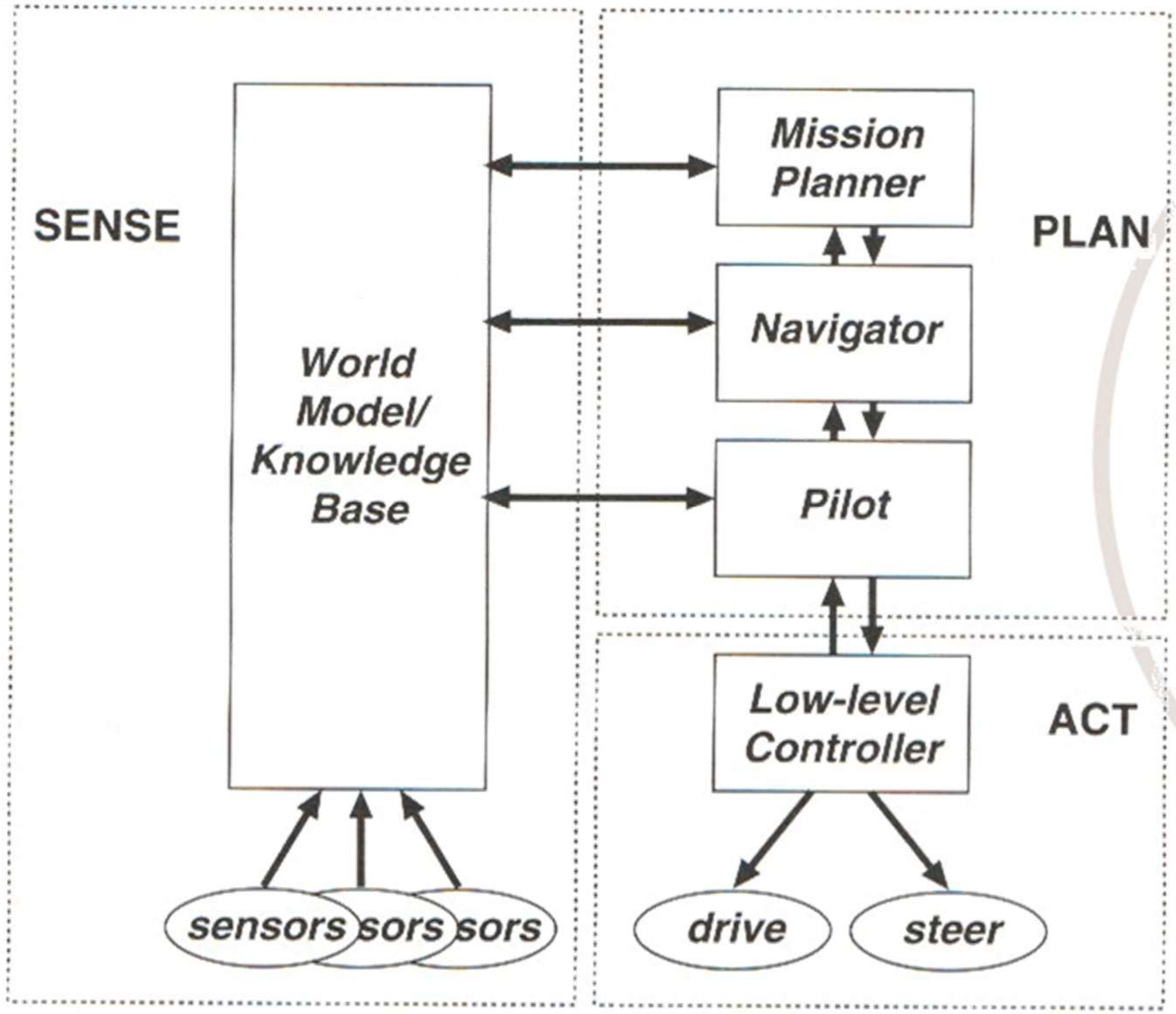
**Command execution**

# Architetture gerarchiche

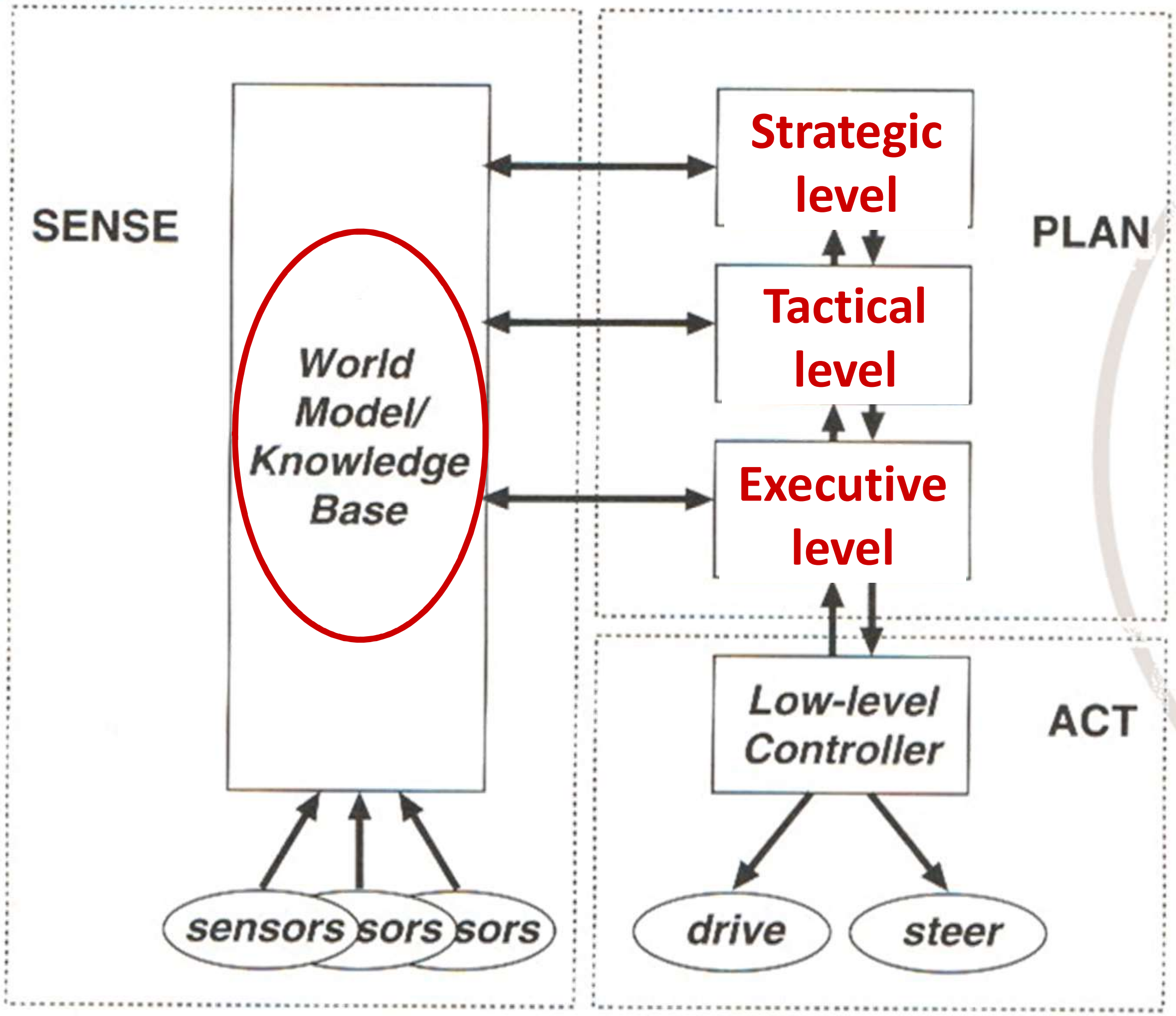
Example for the task: *“take the bottle out of the fridge”*

- **Strategic level**: go to the kitchen, go in front of the fridge, open the fridge, take the bottle…
- **Tactical level**:
  - Go to the kitchen: move_base(X1,Y1); move_base(X2,Y2)…
  - Open the fridge: move_arm(P1), open_hand()….
- **Executive level**:
  - Move_base(X1,Y1); move_base(X2,Y2); move_braccio(P1)…

# Nested Hierarchical Controller



# Nested Hierarchical Controller



# Nested Hierarchical Controller - PLAN

THE BIOROBOTICS INSTITUTE

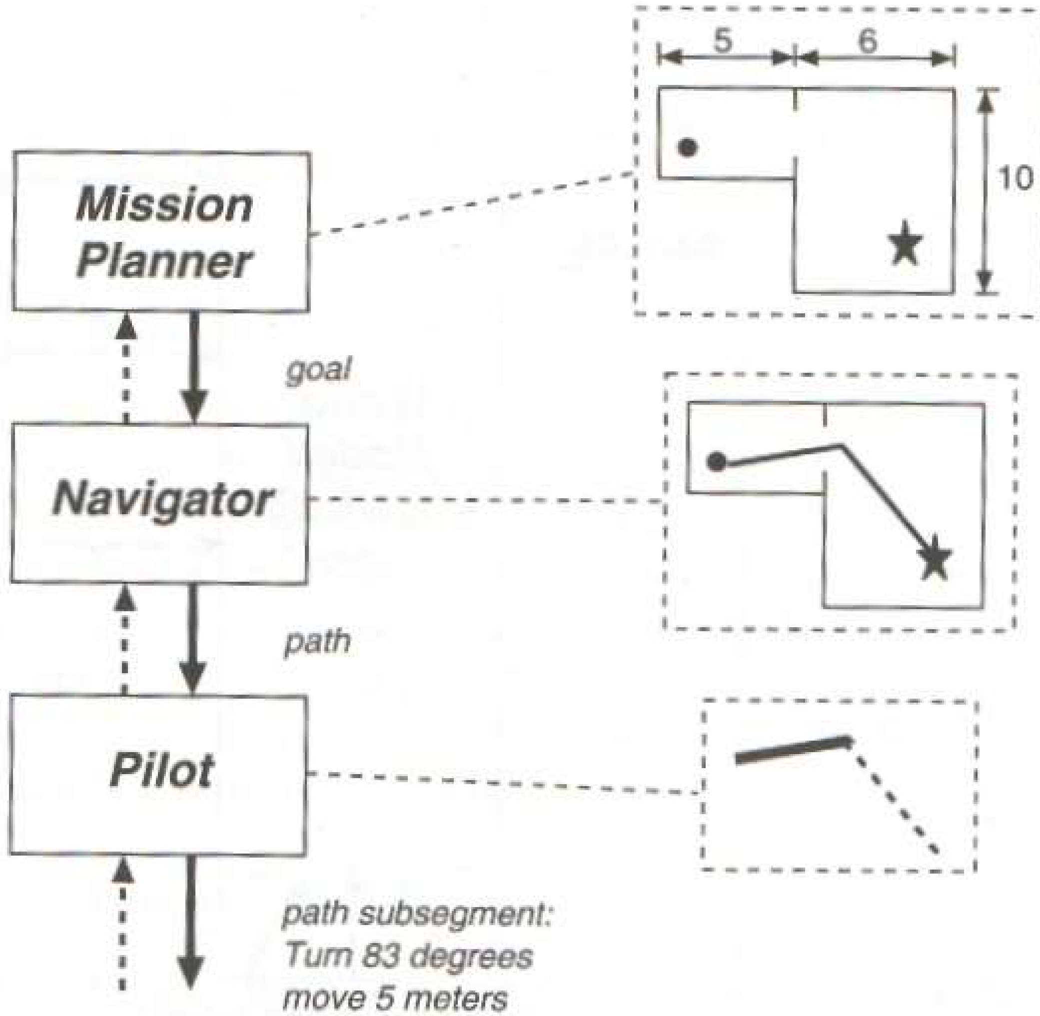



Figure 2.6 Examination of planning components in the NHC architecture.

- The Mission Planner module receives a mission from a human operator (ex. take the boxe in the next room) and encodes it in terms usable by the other modules. It also derives the position and goal of the robot from a map
- The Navigator module receives such information and generates the trajectory from the current position to the goal
- The Pilot modules generates the actions that the actuators have to perform for following the trajectory

# Hierarchical architectures

**Advantages**

- Clear order between perception, planning and action
- Predictable behaviour, e.g. a priori behaviour planning
- System stability

**Disadvantages**

- High computation cost, especially due to the world model update and to planning
- Separation between perception, planning and action and consequent low reactivty, e.g. limited adaptability to real-time environment modifications
- Poor uncertainty management and effectiveness
- Low parallelism

# Hierarchical architectures

## Drawback 1: Time-Scale

*Generating a plan for a real environment can be very slow.*

## Drawback 2: Space

*Generating a plan for a real environment can be very memory-intensive.*

## Drawback 3: Information

*Generating a plan for a real environment requires updating the world model, which takes time.*

## Drawback 4: Use of Plans

*Executing a plan, even when one is available, is not a trivial process.*

# Robot behaviour

Primitive functions

No ‘cognition’ module
Direct interaction between perception and action modules

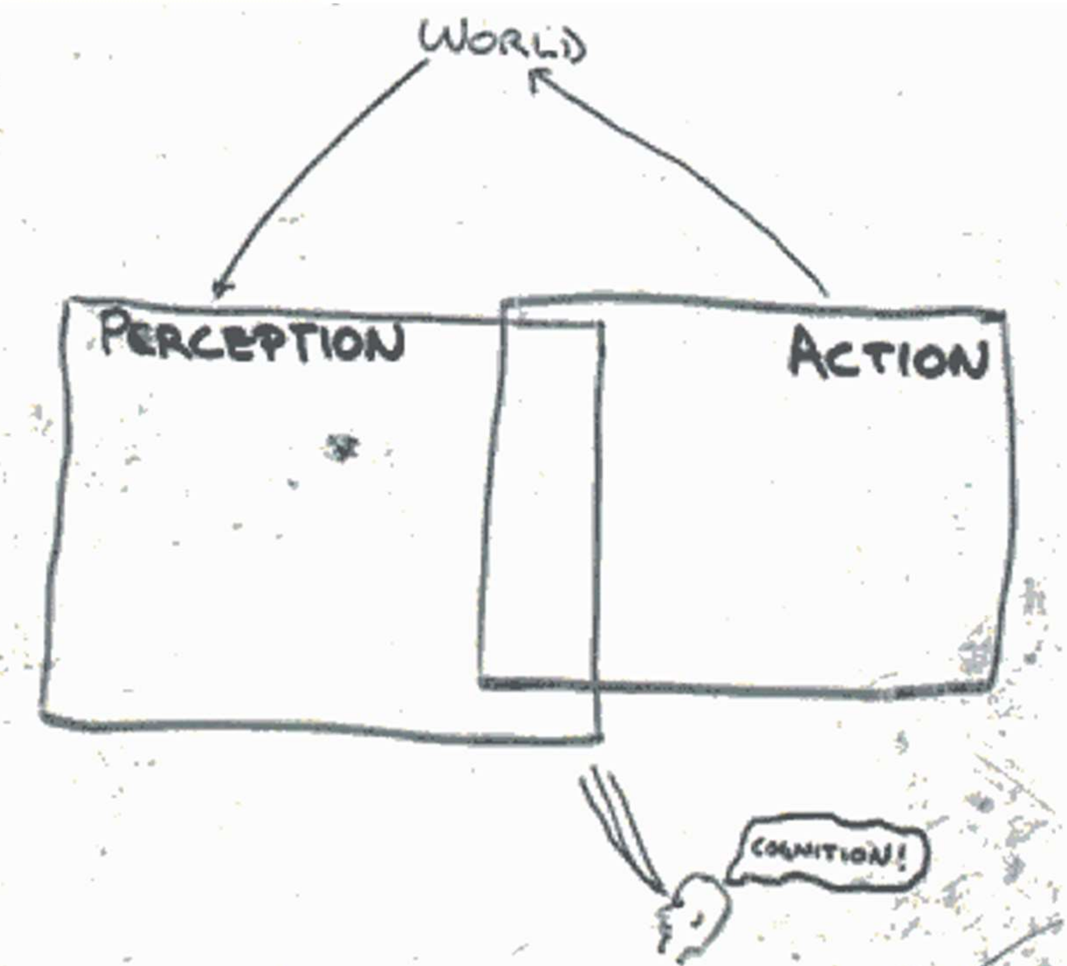
“The world is its own best model” (just need sensors)

“Cognition is in the eyes of the observer”


Reactive architectures

R. Brooks, *Cambrian Intelligence,* MIT Press, 2000

# Reactive architectures or *behavior-based architectures*




No ‘cognition’ module
Direct interaction between perception and action modules

“The world is its own best model”
(just need sensors)

“Cognition is in the eyes of the observer”

R. Brooks, *Cambrian Intelligence,* MIT Press, 2000

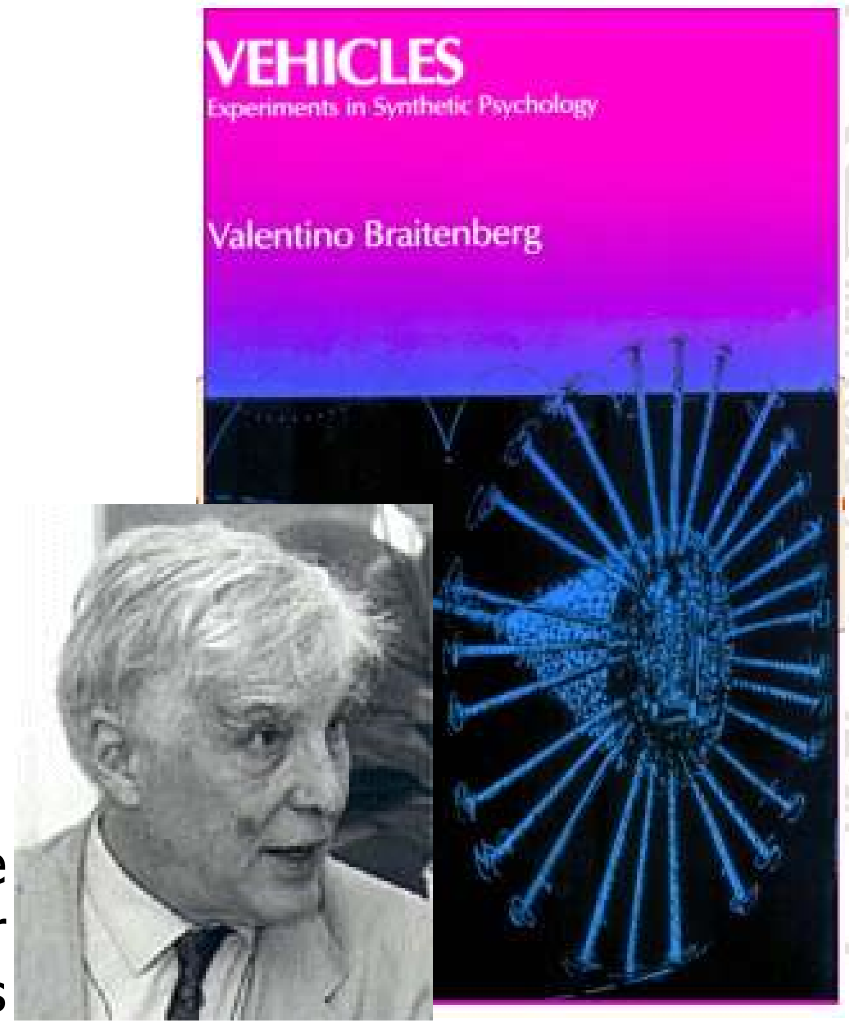
# Reactive paradigm

| ROBOT PRIMITIVES | INPUT | OUTPUT |
|---|---|---|
| SENSE | Sensor data | Sensed information |
| PLAN | Information (sensed and/or cognitive) | Directives |
| ACT | Sensed information or directives | Actuator commands |

# Before Brooks…

**Vehicles**
**Experiment in Synthetic Psychology**

By Valentino Braitenberg
The MIT Press




Director of the
Max Planck Institute For
Biological Cybernetics

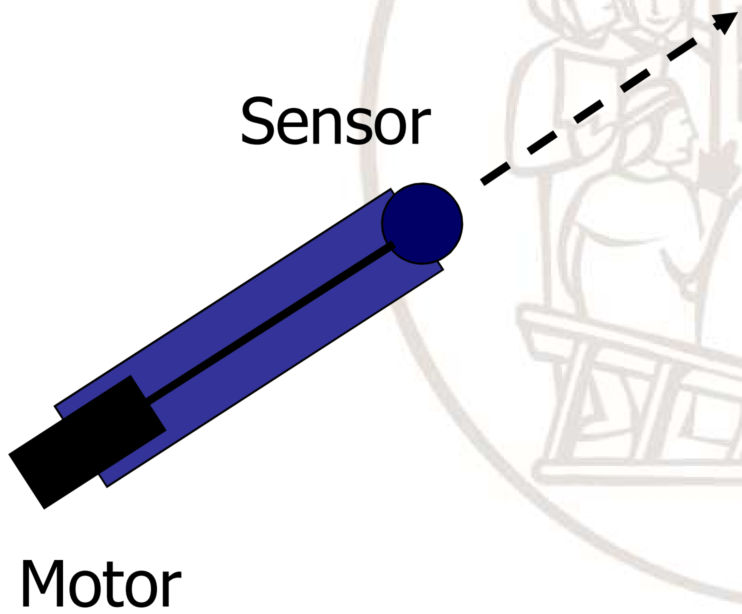
# Experiment 1

THE BIOROBOTICS INSTITUTE

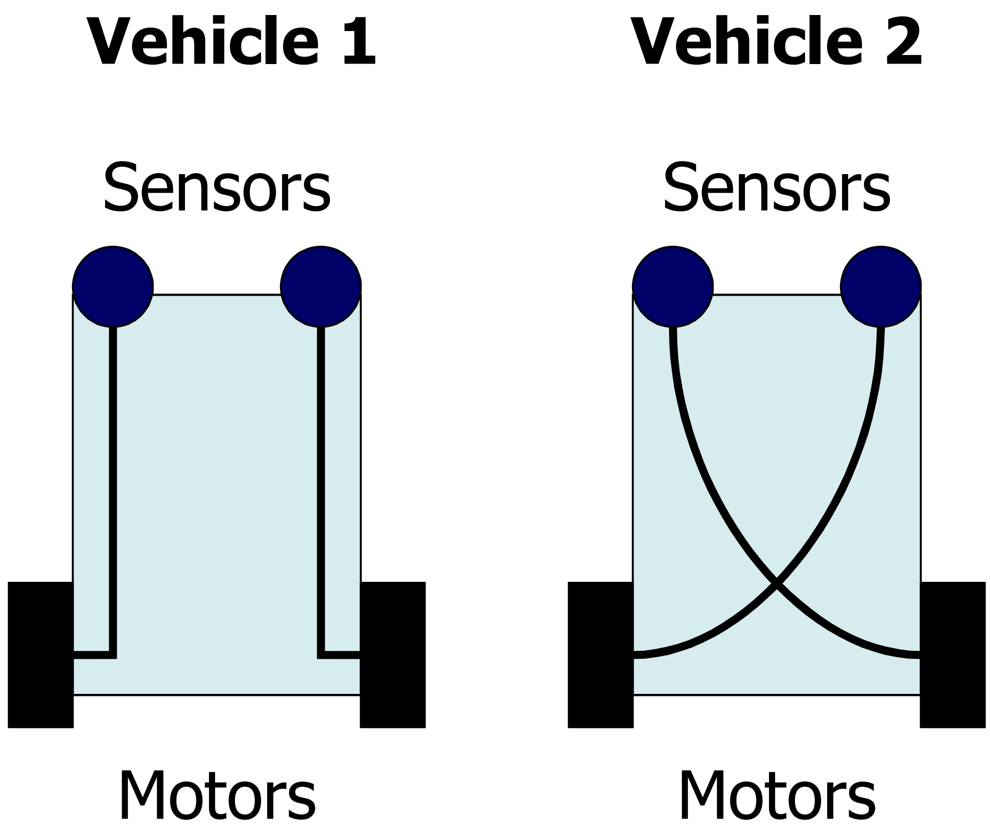
Scuola Superiore Sant'Anna

- Direct connection between sensor and motor
- The motor speed is proportional to the temperature returned by the sensor

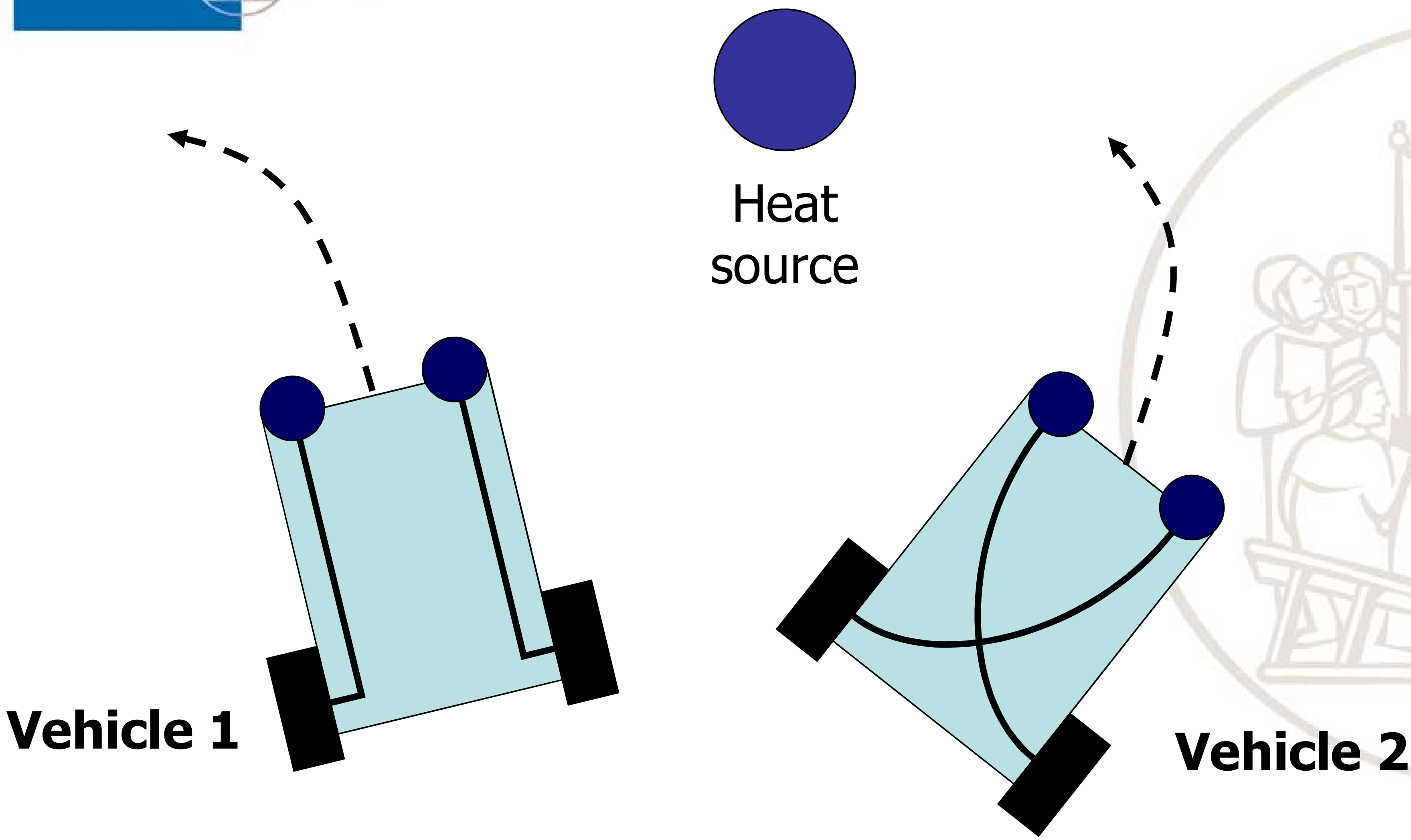
- Resulting behaviour?
- The vehicle moves along a same direction, faster in warmer areas, slowlier in coler areas



# Experiment 2: fear and aggression



# Experiment 2: fear and aggression

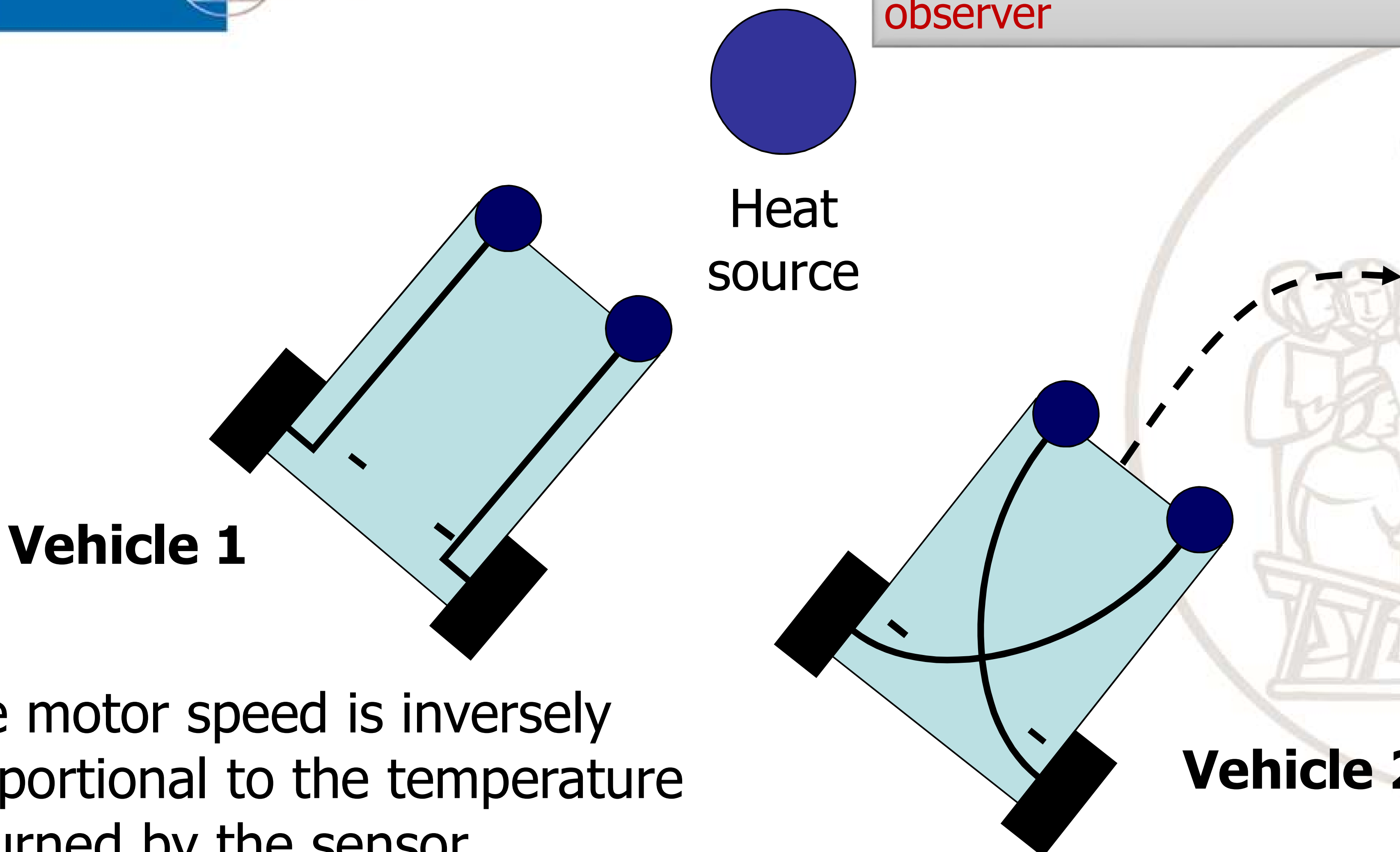


THE BIOROBOTICS INSTITUTE

Scuola Superiore Sant'Anna

# Experiment 3: love

Message: with very simple connections between sensors and actuators, behaviours are obtained that seem 'cognitive' in the eyes of the observer




The motor speed is inversely proportional to the temperature returned by the sensor

# From hierarchical to reactive architectures

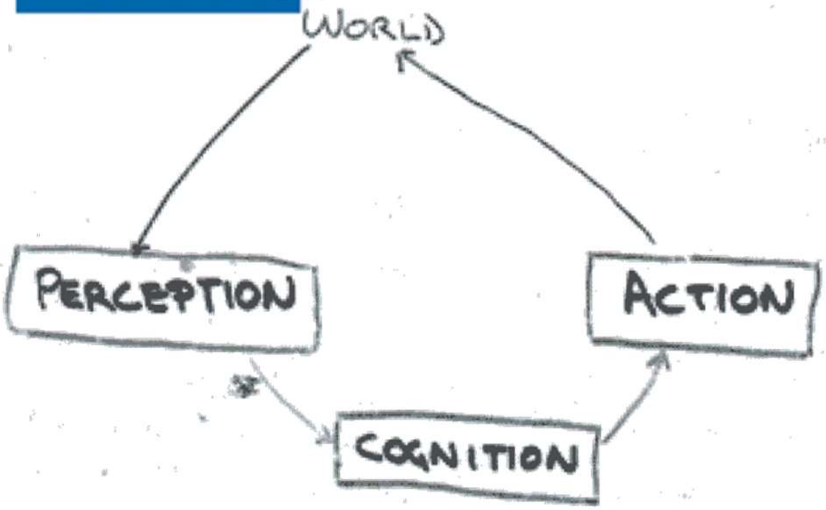



**Figure 1:** The traditional model where cognition mediates between perceptions and plans of actions.

deliberative, model-based

reactive, behavior-based

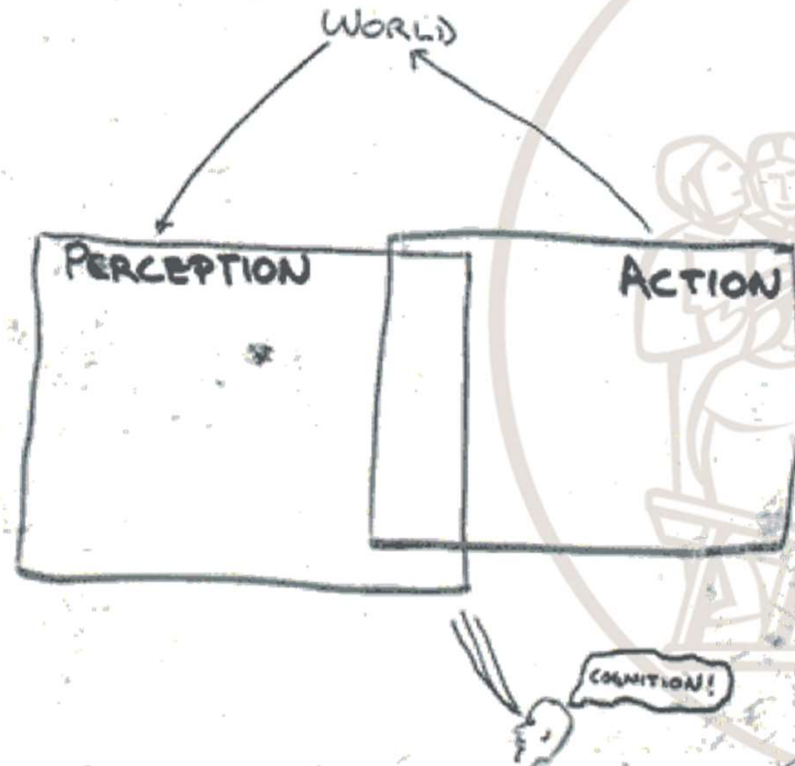


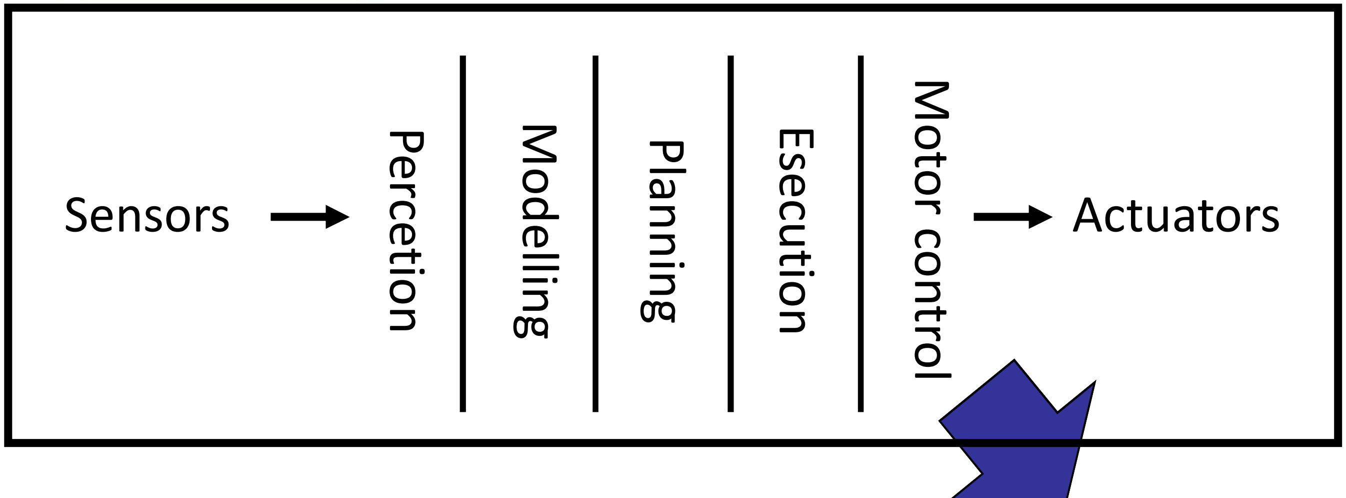

**Figure 2:** The new model, where the perceptual and action subsystems are all there really is. Cognition is only in the eye of an observer.

*R. Brooks, "Cambrian Intelligence", MIT Press, 1999*

# From hierarchical to reactive architectures

THE BIOROBOTICS INSTITUTE

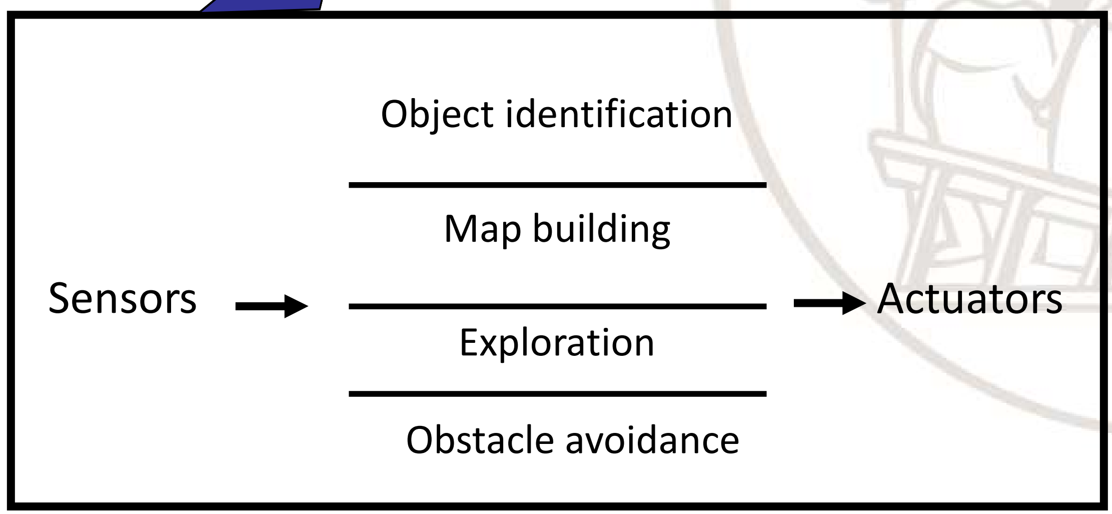
Scuola Superiore Sant'Anna



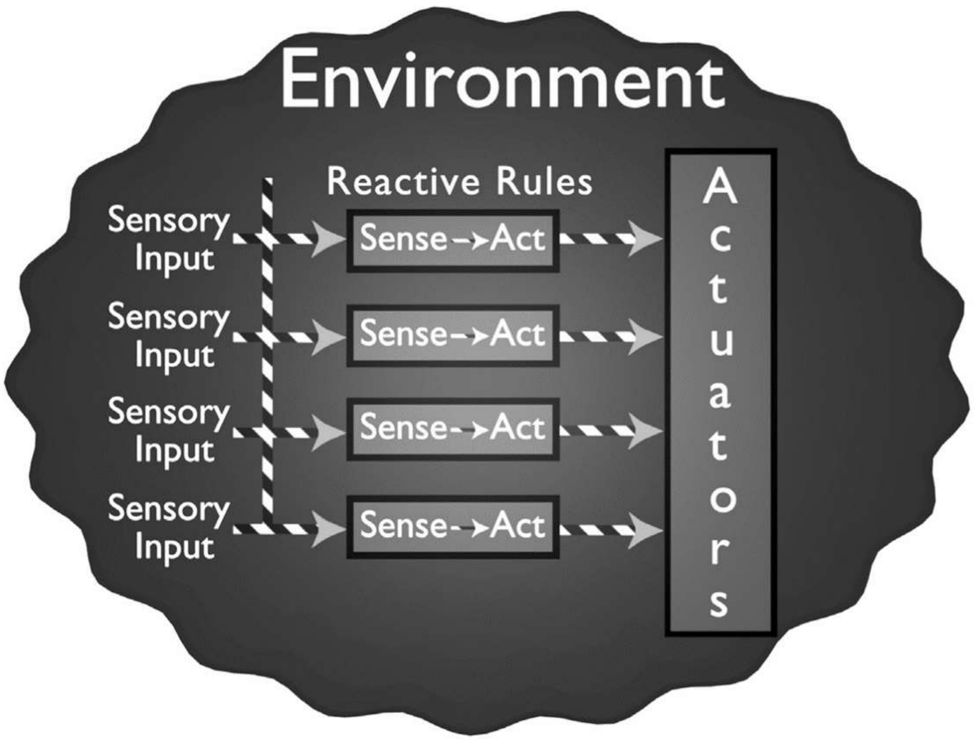

From *horizontal and sequential* division of the information processing chain to *vertical and parallel* division

***Competence levels***

Decomposition based on desired internal manifestation, not based on internal robot working



# Reactive architectures




Maja J Mataric, *The Robotics Primer*, The MIT Press, 2007

# Reactive architectures
# Example:

Suppose that you are asked to write a reactive controller that will enable a robot to move around and avoid obstacles. The robot has two simple whiskers, one on the left and one on the right. Each whisker returns 1 bit, "on" or "off"; "on" indicates contact with a surface (i.e., the whisker is bent).

```
If left whisker bent, turn right.
If right whisker bent, turn left.
If both whiskers bent, back up and turn to the left.
Otherwise, keep going.
```

*A robot using the above controller could oscillate if it gets itself into a corner where the two whiskers alternate in touching the walls.*

# Reactive architectures
# Example:

Now suppose that instead of just two whiskers, your robot has a ring of sonars (twelve of them, to cover the 360-degree span, as you learned in Chapter 9). The sonars are labeled from 1 to 12. Sonars 11, 12, 1 and 2 are at the front of the robot, sonars 3 and 4 are on the right side of the robot, sonars 6 and 7 are in the back, and sonars 1 and 10 are on the left

```
(case
  (if (minimum (sonars 11 12 1 2))   <=   danger-zone
         and
       (not stopped)
  then
    stop)
(if ((minimum (sonars 11 12 1 2))    <=  danger-zone
         and
       stopped)
then
  move backward)
(otherwise
  move forward))
```


Maja J Mataric, *The Robotics Primer*, The MIT Press, 2007

# Reactive architectures

# Example:

### Adding a module:

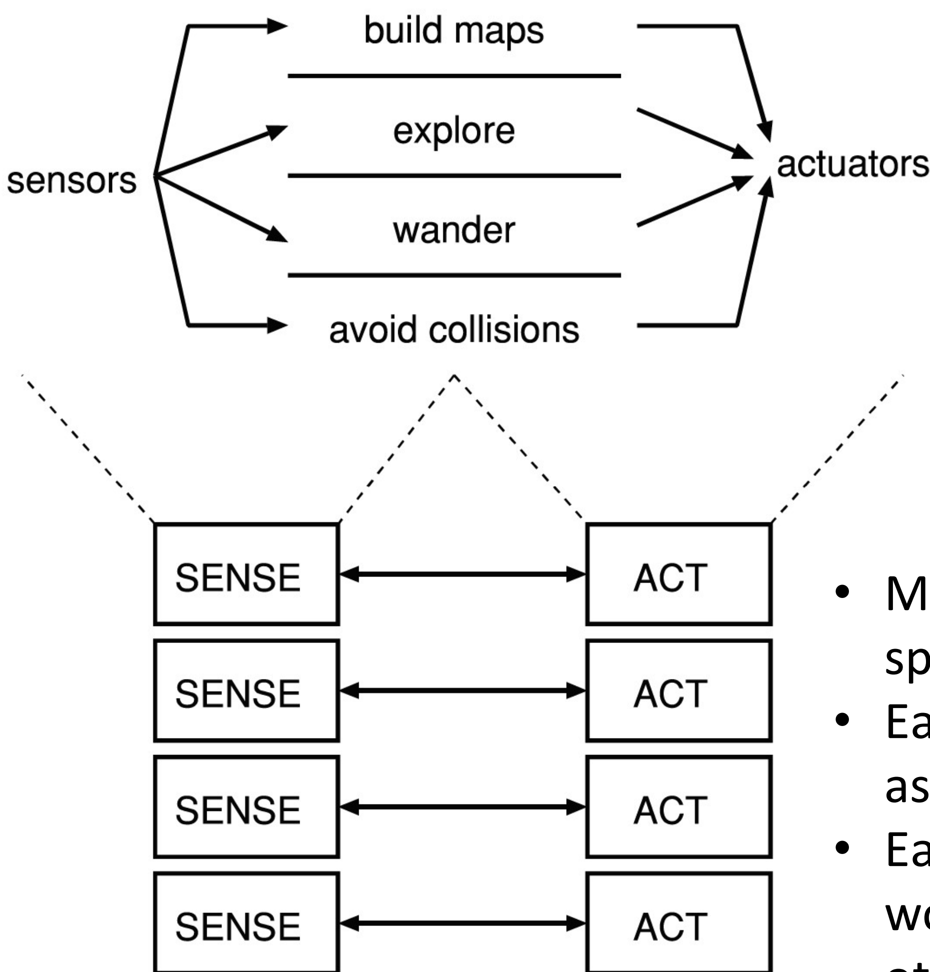
```
(case
  (if ((sonar 11 or 12) <= safe-zone
      and
    (sonar 1 or 2) <= safe-zone)
  then
    turn left)
  (if (sonar 3 or 4) <= safe-zone
  then
    turn right))
```


The above controller makes the robot turn away from detected obstacles. Since safe-zone is larger than danger-zone, this allows the robot to turn away gradually before getting too close to an obstacle and having to be forced to stop, as in the previous controller. If obstacles are detected on both sides, the robot consistently turns to the left, to avoid oscillations.

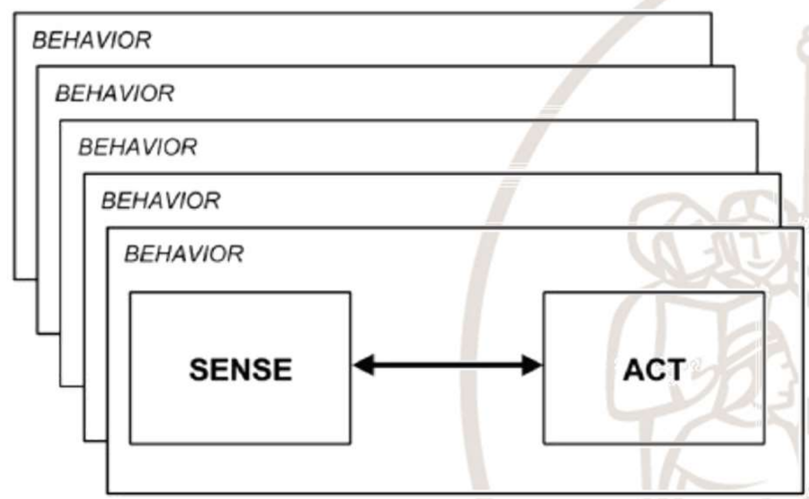
By combining the two controllers above we get a wandering behavior which avoids obstacles at a safe distance while moving smoothly around them, and also avoids collisions with unanticipated nearby obstacles by stopping and backing up.

# Reactive architectures or *behavior-based architectures*






The SENSE-ACT couple is named BEHAVIOUR




- Multiple information flows, each related to a specific robot function
- Each behaviour is concerned with one specific aspect of the overall behaviour
- Each behaviour is a finite-state machine ad it works asynchronously and in parallel with the others

# Reactive architectures or *behavior-based architectures*

- The robot behaviours are reactions to the information perceived from the environment
- The basic module is a so-called **behaviour**, obtained from a direct interaction between sensors and actions
- The robots based on reactive architectures are called **reactive robots**, i.e. robots responding to environmental stimuli in real-time, and the term ***behaviour-based robotics*** is also used.

# Reactive architectures or *behavior-based architectures*

THE BIOROBOTICS INSTITUTE

Scuola Superiore Sant'Anna

- The robot interacts with the environment with sensors and actuators
- **There is no world representation** ("The world is its best model", R. A. Brooks, 1986): the knowledge on the world is not modelled nor stored in a memory, but it is extracted in real time from the world itself, through sensors
- Since a world model does not exist, a priori planning of the robot actions cannot exist

# Reactive architectures or *behavior-based architectures*

1. **Situated agent**: the robot is a situated agent operating in an ecological niche. It is an integral part of the world and when it acts it changes the world and receives new sensory inputs.
2. **Behaviour-based**: behaviours serve as the basic building blocks for robotic actions, and the overall behaviour of the robot is emergent. Behaviours are independent, computational entities and operate concurrently.
3. **Locality**: only local, behaviour-specific sensing is permitted. The use of explicit abstract representational knowledge in perceptual processing, even though it is behaviour-specific, is avoided.
4. **Independence**: the various behaviours must be independent to each other. As a consequence, a shared world model is not possible.

# Reactive architectures or *behavior-based architectures*

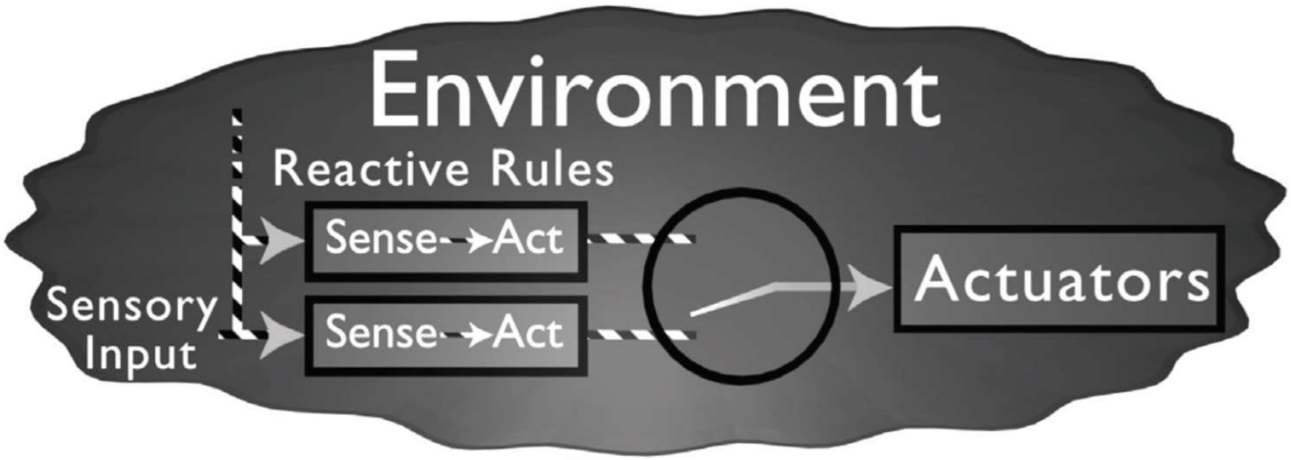
**Advantages**

- High adaptability to environment changes (real-time response)
- Low computational complexity in each behaviour and the overall computational cost is low
- Parallelism
- Extension of behaviours is very easy thanks to modularity
- No world model

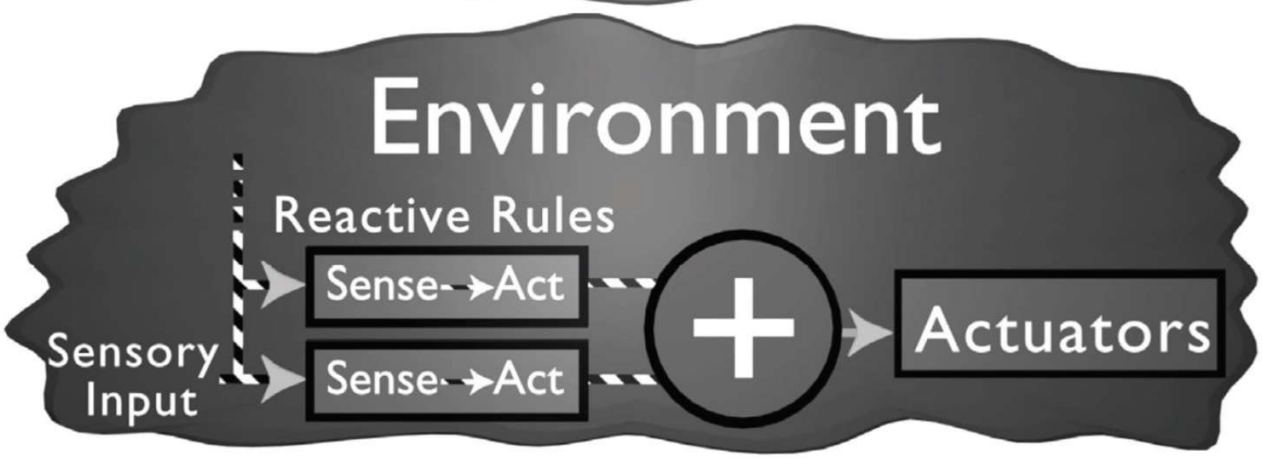
**Disadvantages**

- The overall robot behaviour is difficult to predict
- Management of concurrency between behaviours
- When increasing the number of behaviours, the complexity of concurrency management also increases, with a consequence difficulty in conflict resolution

# Action selection




**Arbitration**




**Fusion**

# An example of reactive architecture: subsumption architecture

THE BIOROBOTICS INSTITUTE

Scuola Superiore Sant'Anna

Collision-free navigation of a mobile robot equipped with ultrasound sensors



## A Robust Layered Control System For A Mobile Robot

RODNEY A. BROOKS, MEMBER, IEEE

***Abstract*—A new architecture for controlling mobile robots is described. Layers of control system are built to let the robot operate at increasing levels of competence. Layers are made up of asynchronous modules that communicate over low-bandwidth channels. Each module is an instance of a fairly simple computational machine. Higher-level layers can subsume the roles of lower levels by suppressing their outputs. However, lower levels continue to function as higher levels are added. The result is a robust and flexible robot control system. The system has been used to control a mobile robot wandering around unconstrained laboratory areas and computer machine rooms. Eventually it is intended to control a robot that wanders the office areas of our laboratory, building maps of its surroundings using an onboard arm to perform simple tasks.**

### I. Introduction

A CONTROL SYSTEM for a completely autonomous mobile robot must perform many complex information processing tasks in real time. It operates in an environment where the boundary conditions (viewing the instantaneous control problem in a classical control theory formulation) are changing rapidly. In fact the determination of those boundary conditions is done over very noisy channels since there is no straightforward mapping between sensors (e.g. TV cameras) and the form required of the boundary conditions.

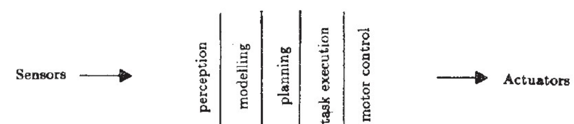



Fig. 1. Traditional decomposition of a mobile robot control system into functional modules.

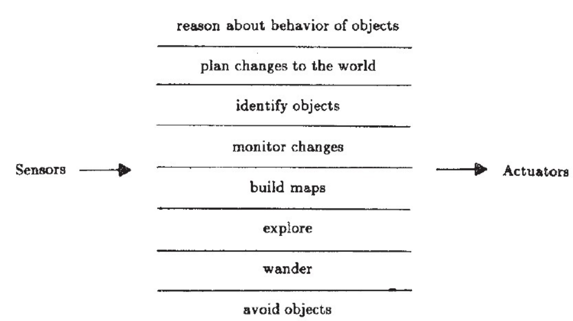
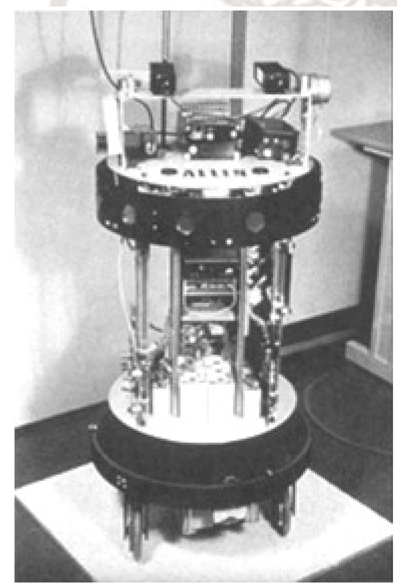



Fig. 2. Decomposition of a mobile robot control system based on task-achieving behaviors.

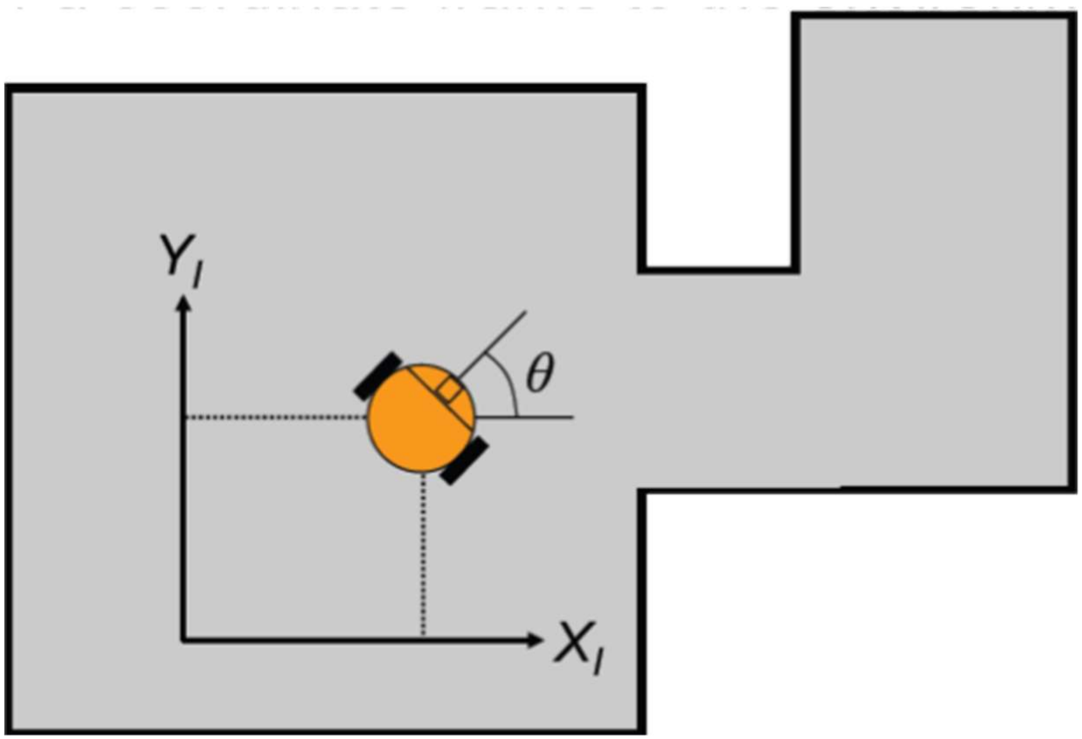


R.A. Brooks, "A Robust Layered Control System for a Mobile Robot", in *Cambrian Intelligence*, The MIT Press, 1999

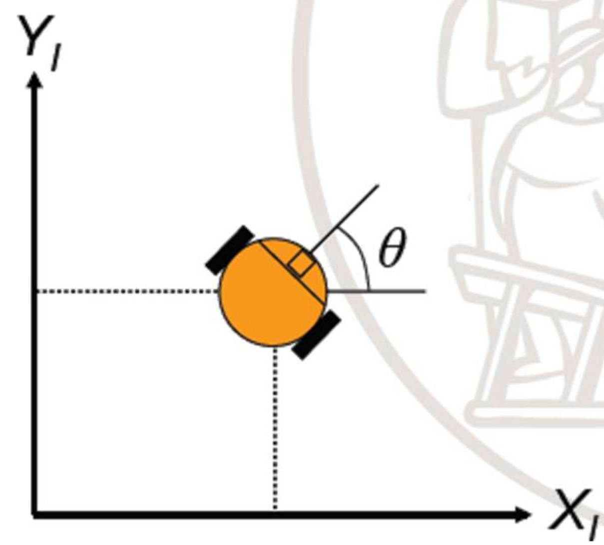
R.A. Brooks, "A Robust Layered Control System for a Mobile Robot", *IEEE Journal of Robotics and Automation*, Vol. Ra-2, No. I, March 1986

# Position of a mobile robot

Reference coordinate system fixed in the world

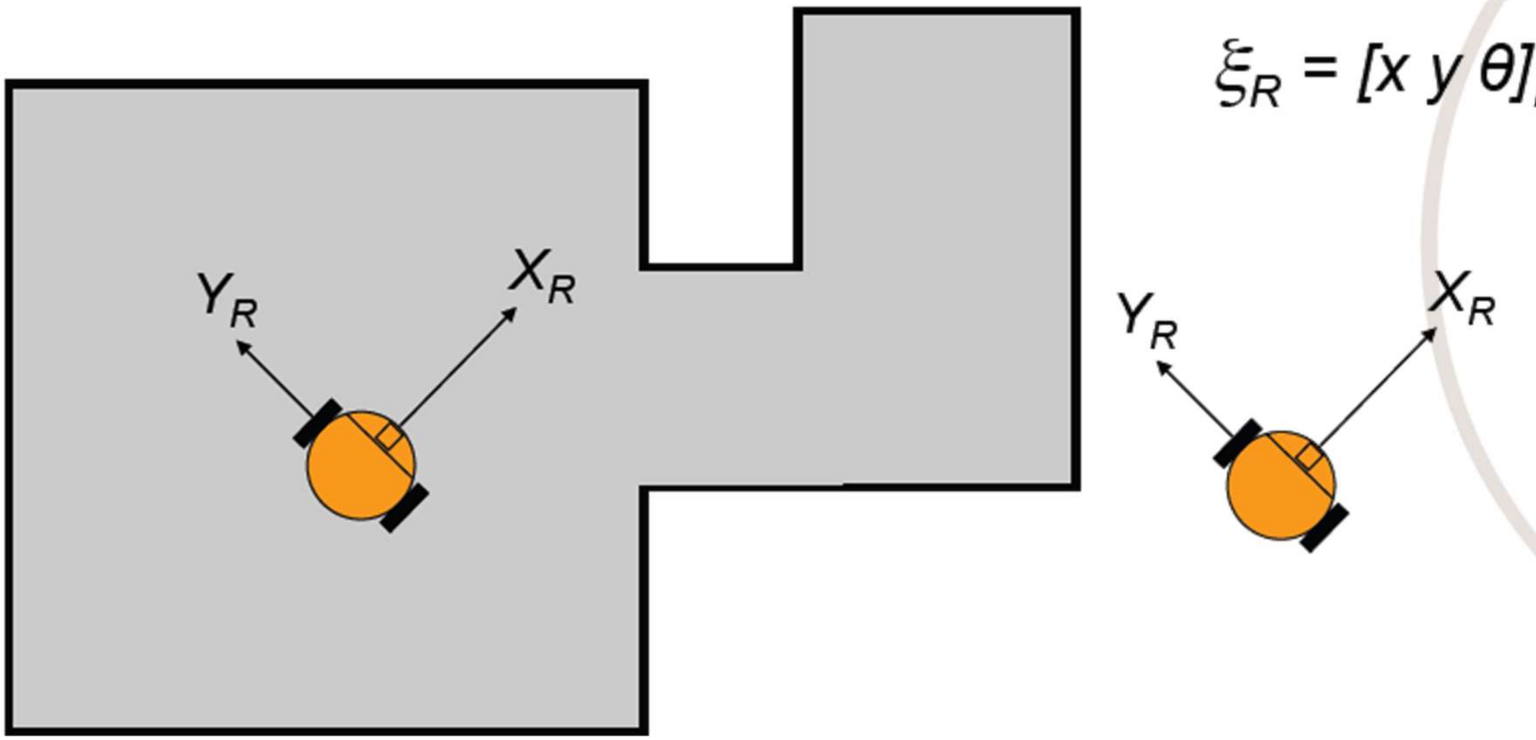



$$\xi_I = [x\ y\ \theta]_I$$



# Position of a mobile robot

Reference coordinate system fixed on the robot

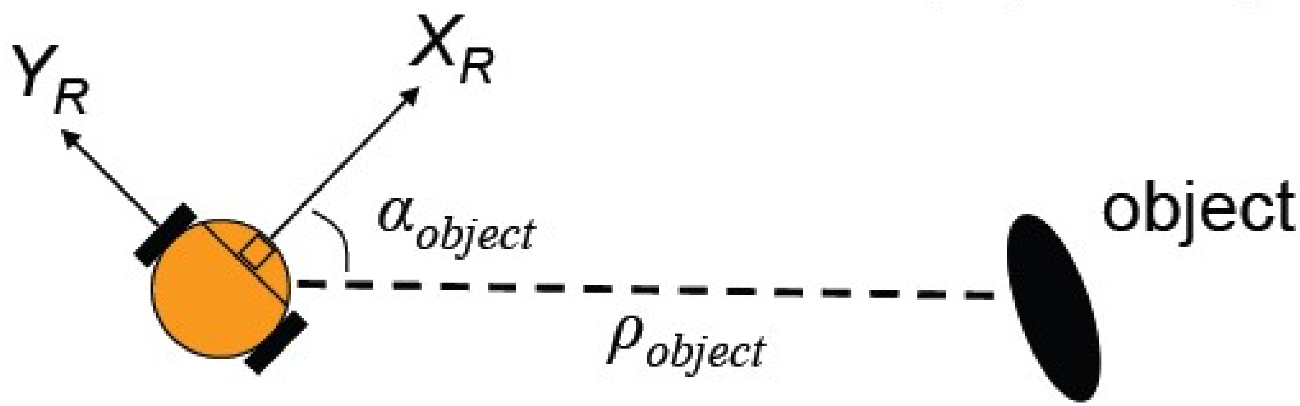



$$\xi_R = [x\ y\ \theta]_R = [0\ 0\ 0]$$

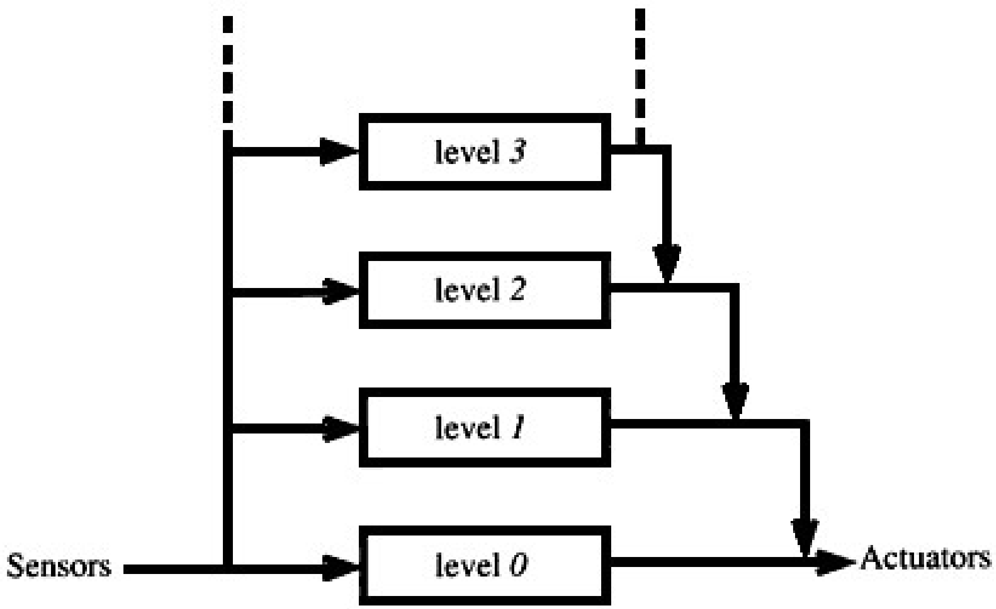
# Position of an obstacle

$$x_{object,R} = \rho_{object} \cos(\alpha_{object})$$

$$y_{object,R} = \rho_{object} \sin(\alpha_{object})$$



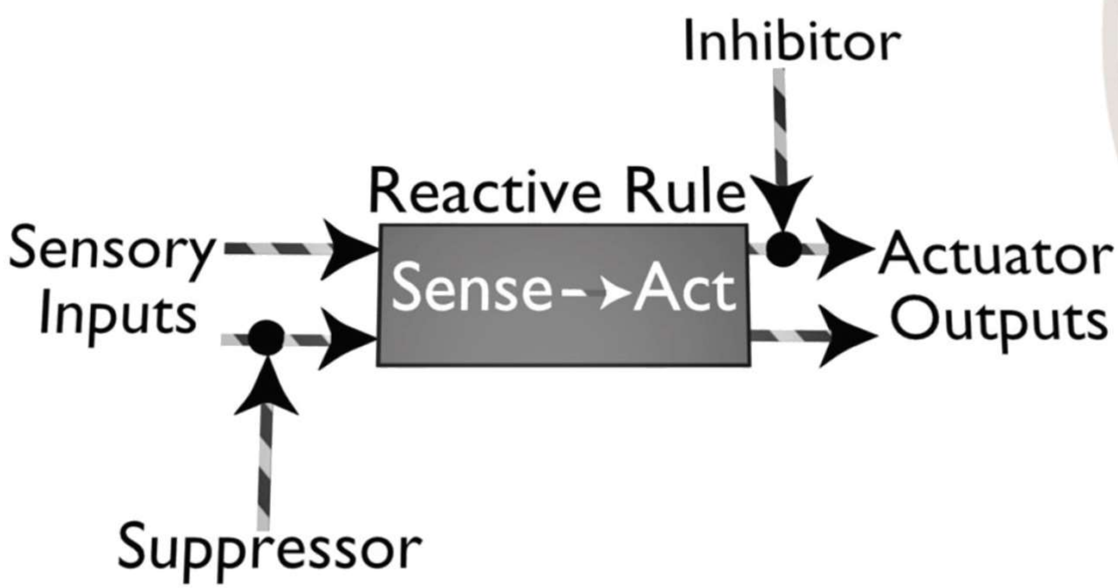
# Subsumption architecture



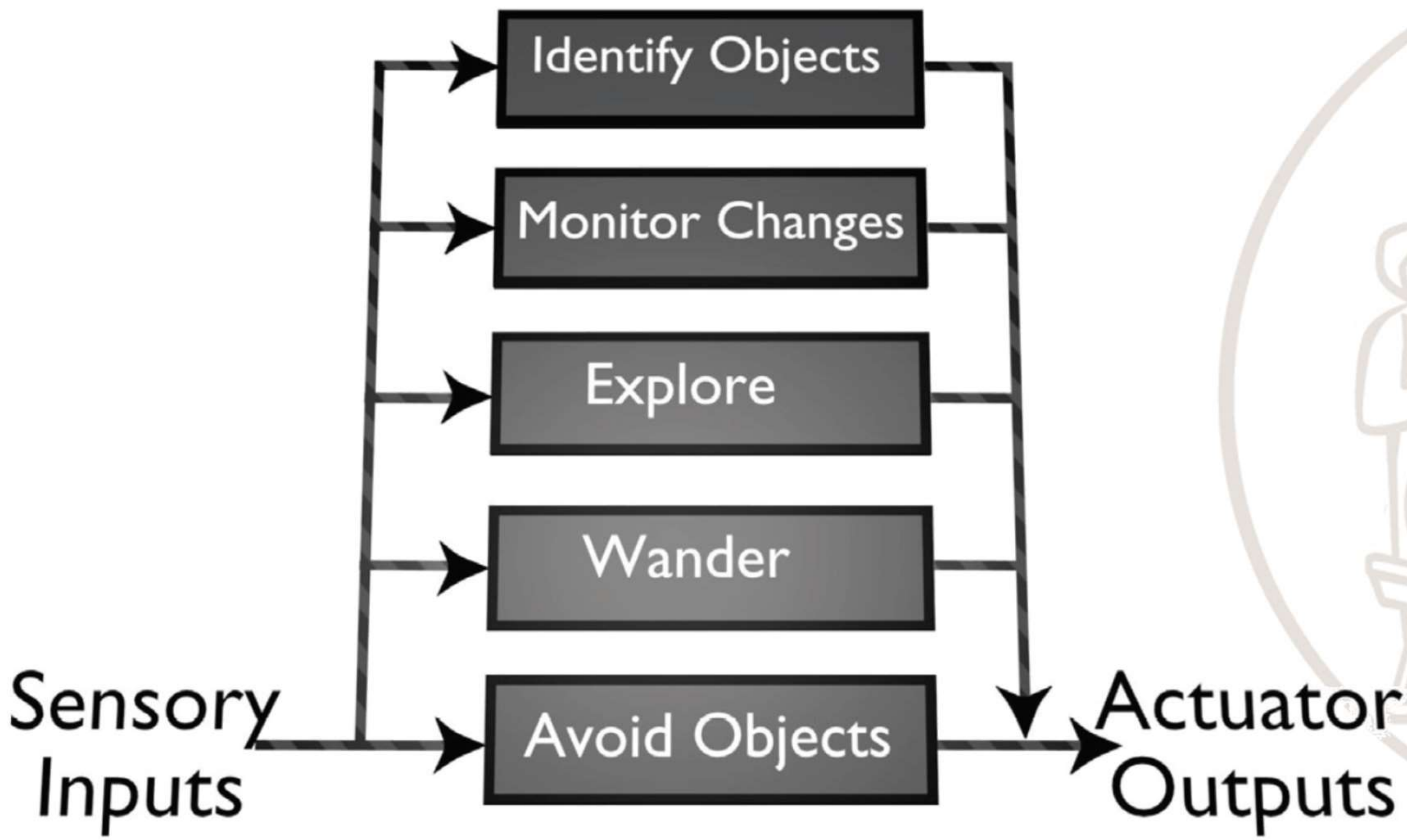

- Behaviours are organized in an architecture based on levels: control levels corresponding to the competence levels of vertical decomposition
  - Lower levels concern more basic functions, like obstacle avoidance
  - Higher levels concern more goal-directed actions.
- Higher levels ‘subsume’ lower levels
- The levels work in an independent and concurrent way

# Subsumption architecture: suppression and inhibition

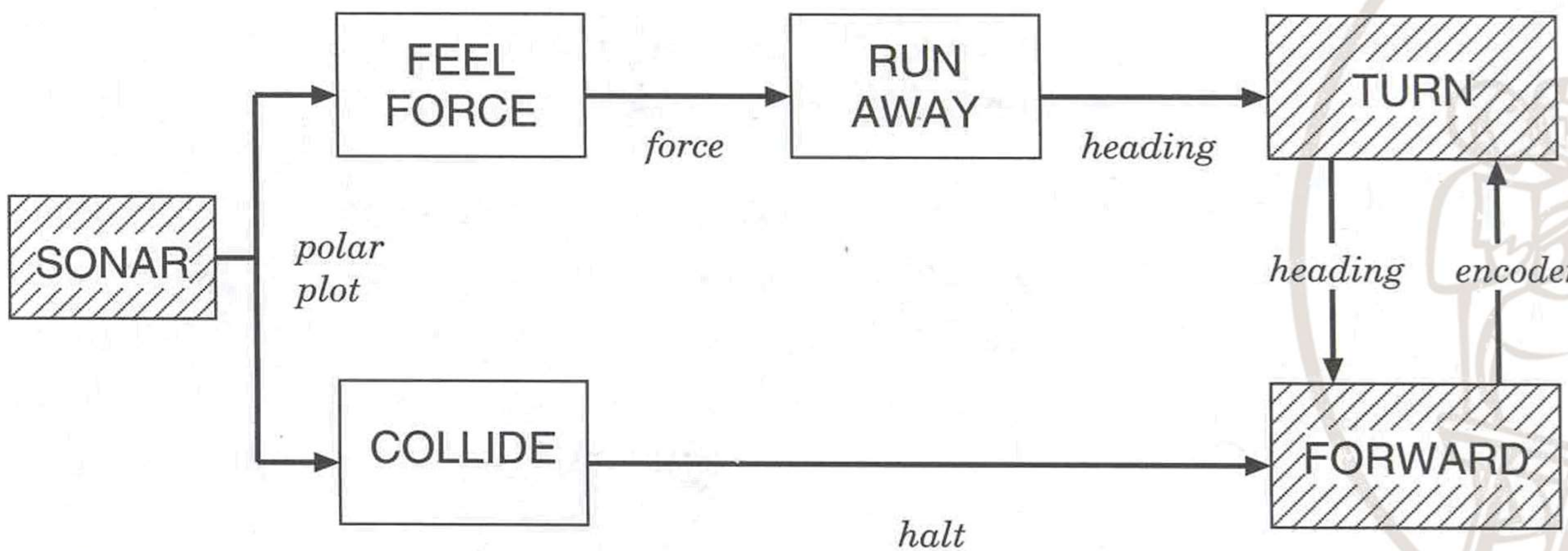
- Each behaviour has input and output lines.
- Outpur lines of a behaviour can be connected to input or output lines of other behaviours:
  - An input signal can be **suppressed** and replaced with the signal that suppressed it
  - An output signal can be **inhibited**



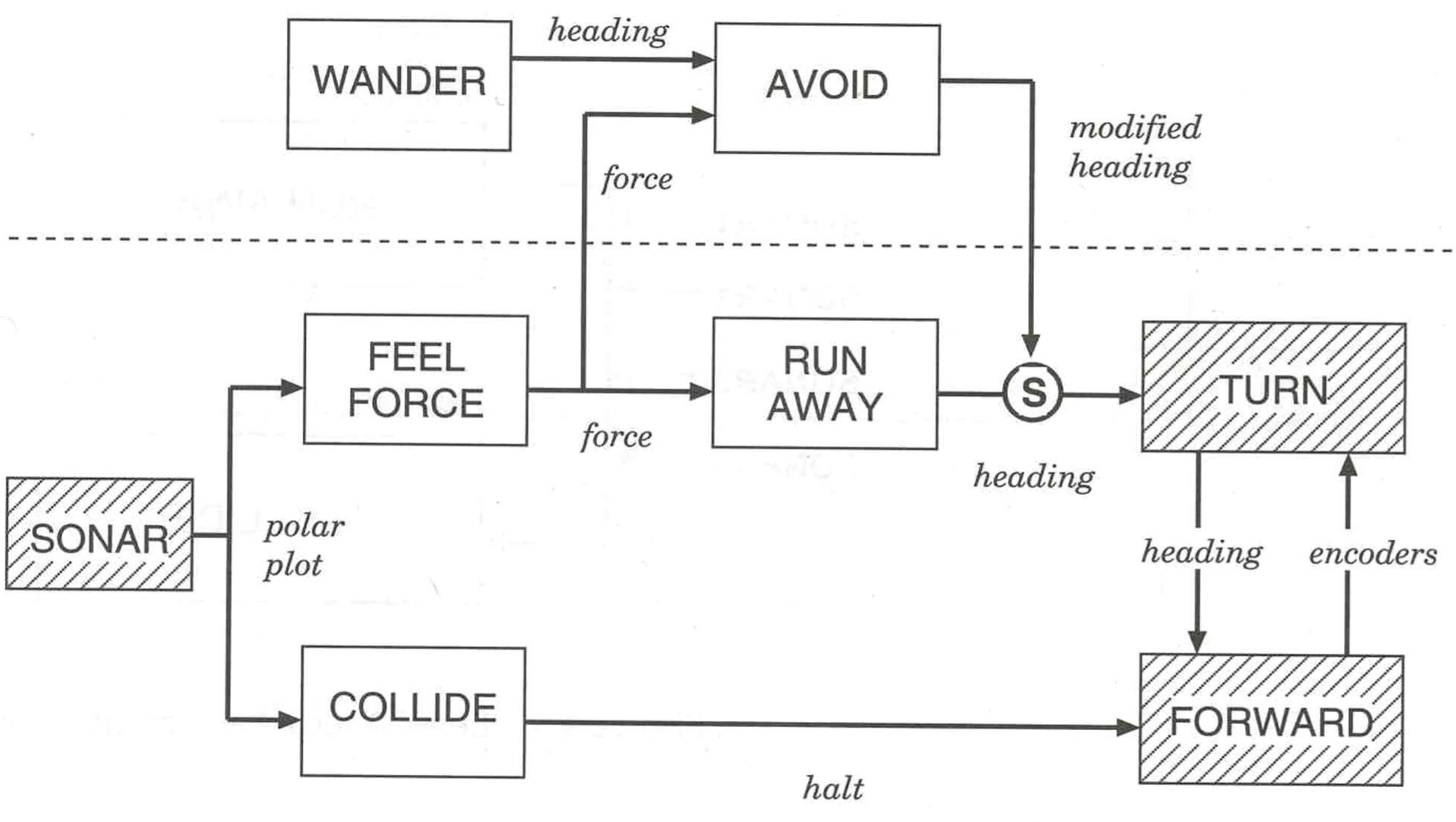
# Subsumption architecture
## for a case of robot navigation



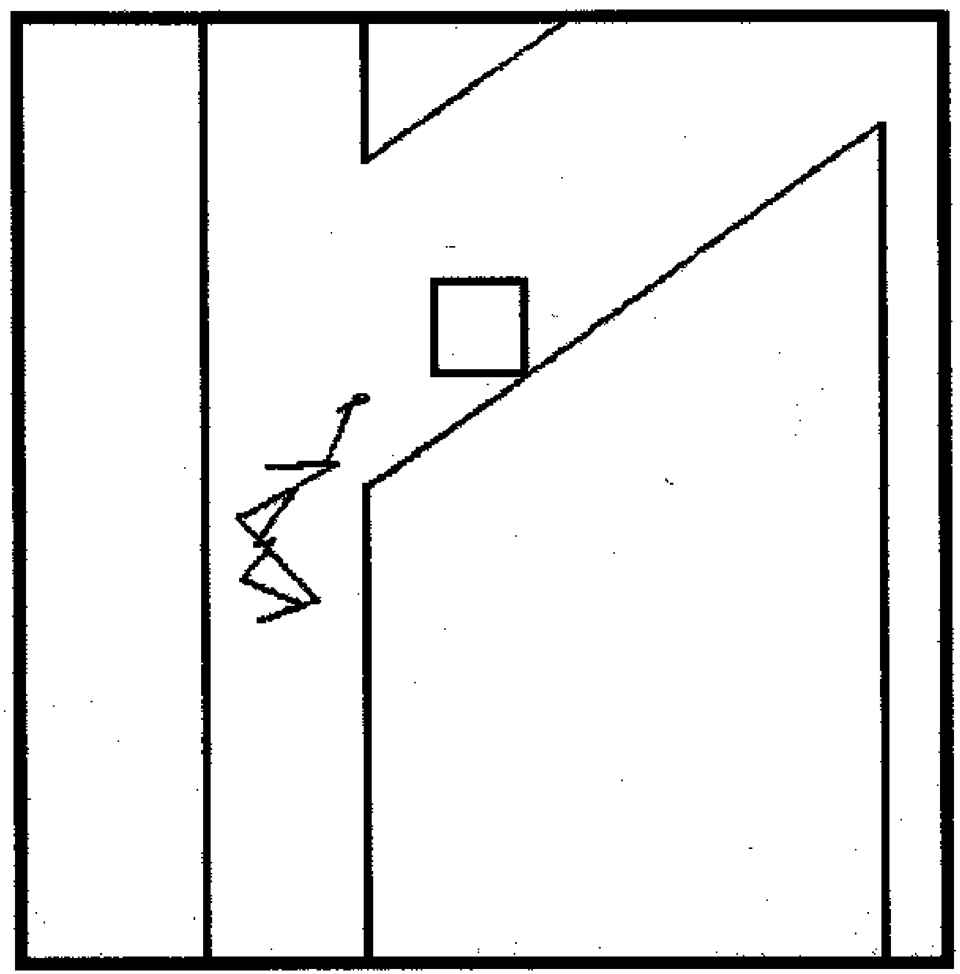
# Level 0 - Avoid



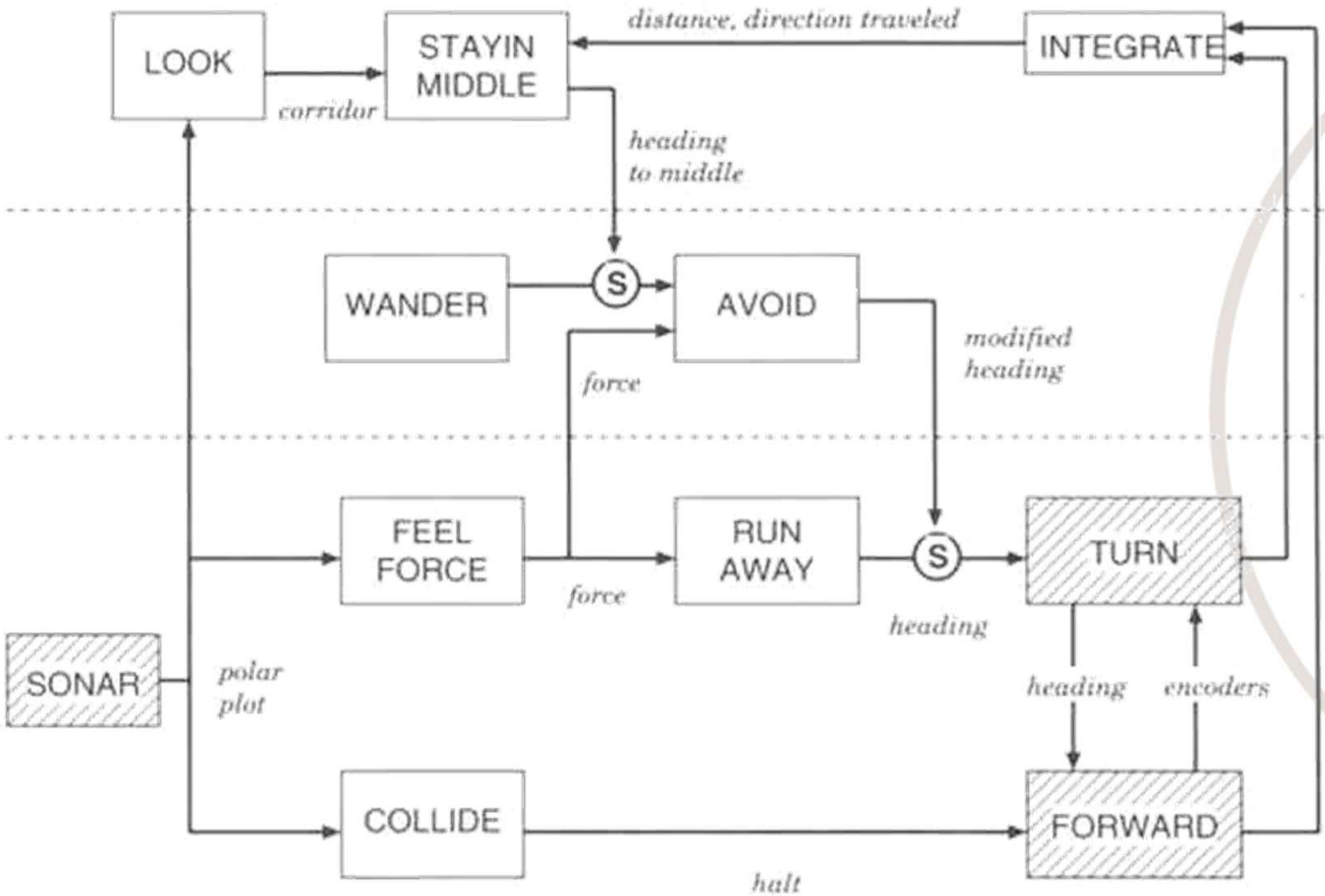
# Level 1 - Wander



# Results from simulations of levels 0 e 1



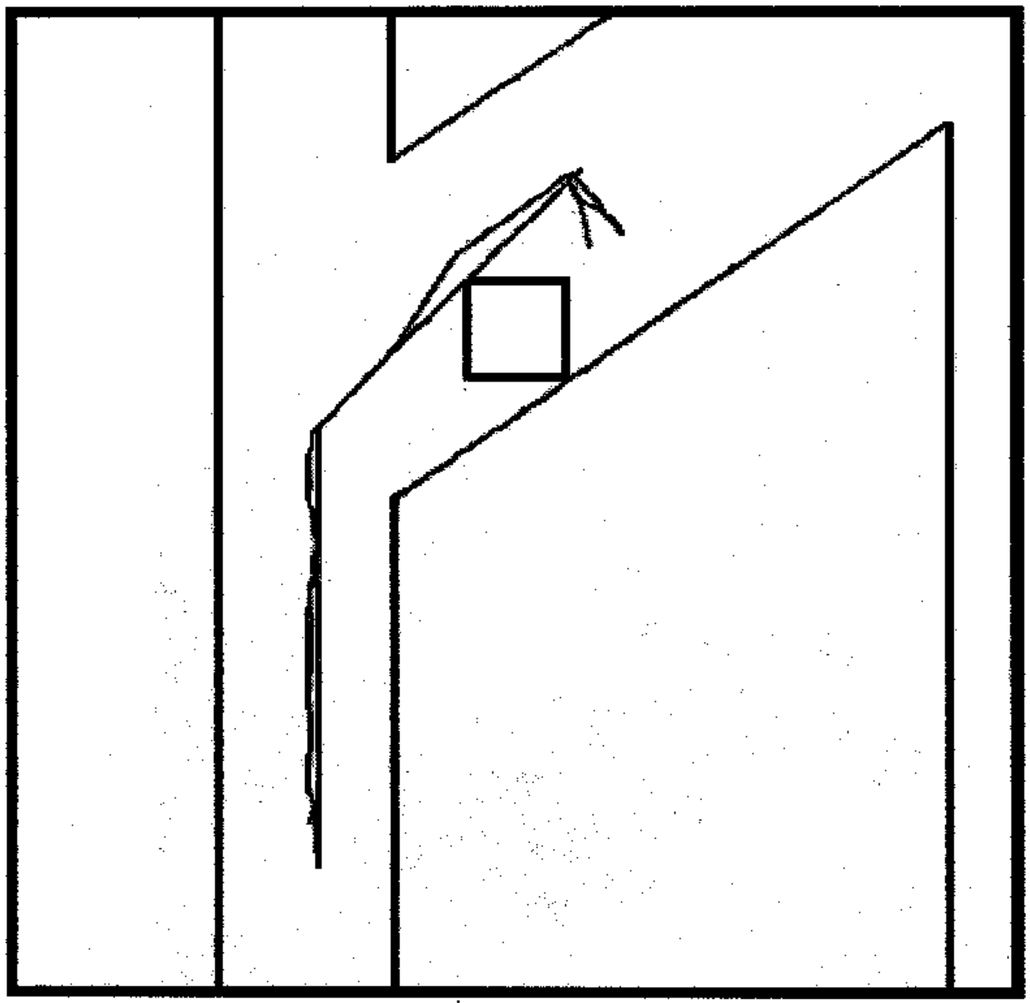
# Level 2 - Explore



THE BIOROBOTICS INSTITUTE

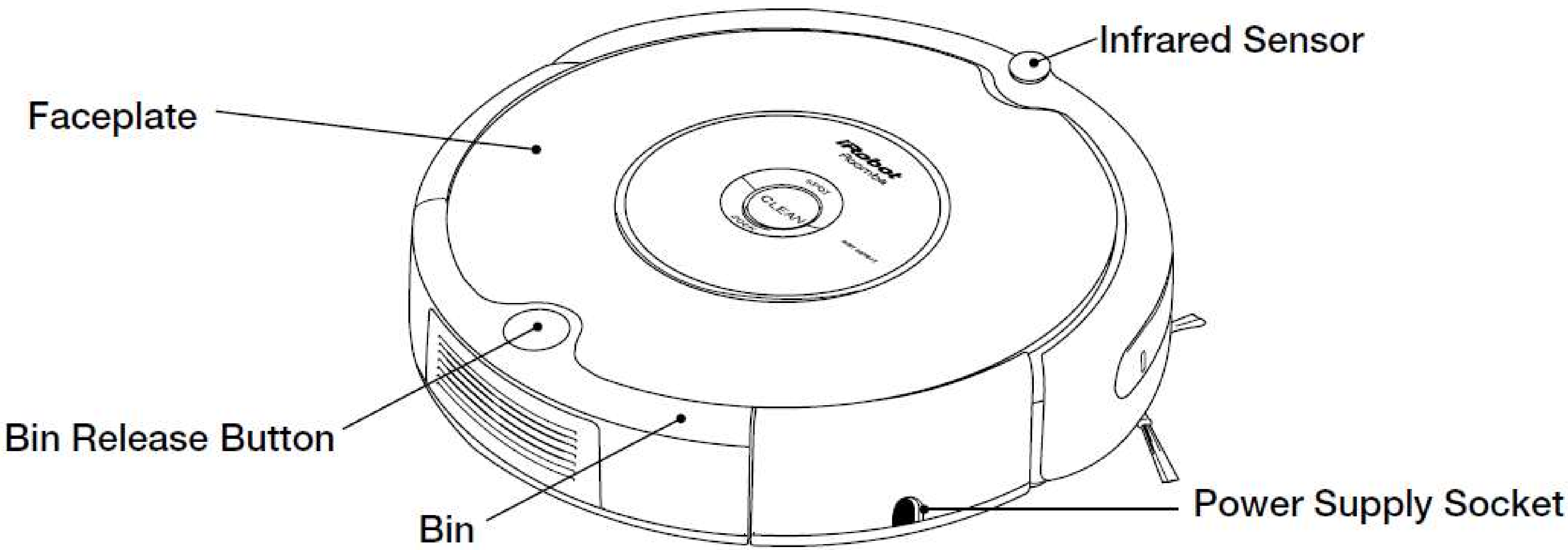
Scuola Superiore Sant'Anna

# Results from simulation of levels 0, 1 e 2



THE BIOROBOTICS INSTITUTE

Scuola Superiore Sant'Anna

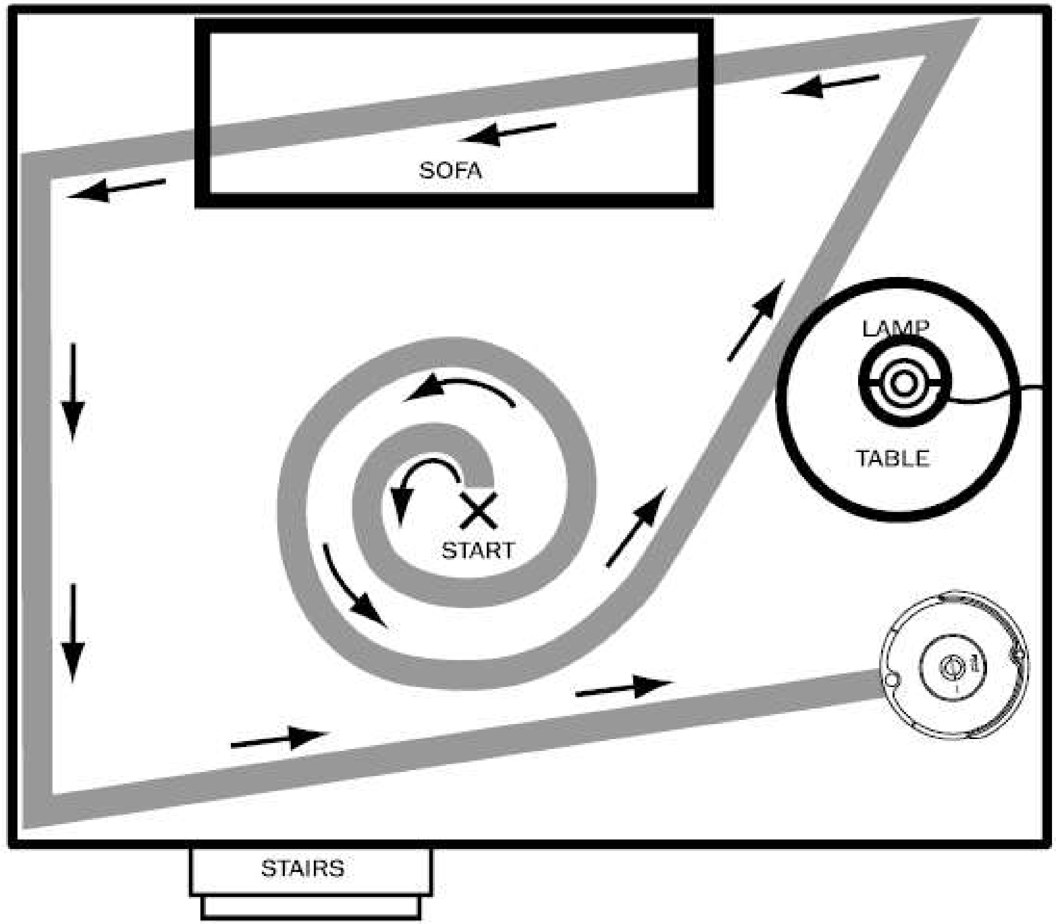
# iRobot Roomba – reactive behaviours

**Spiraling:** Roomba uses a spiral motion to clean a concentrated area.

**Wall Following:** Roomba uses this technique to clean the full perimeter of the room and navigate around furniture and obstacles.

**Room Crossing:** Roomba crisscrosses the room to ensure full cleaning coverage.

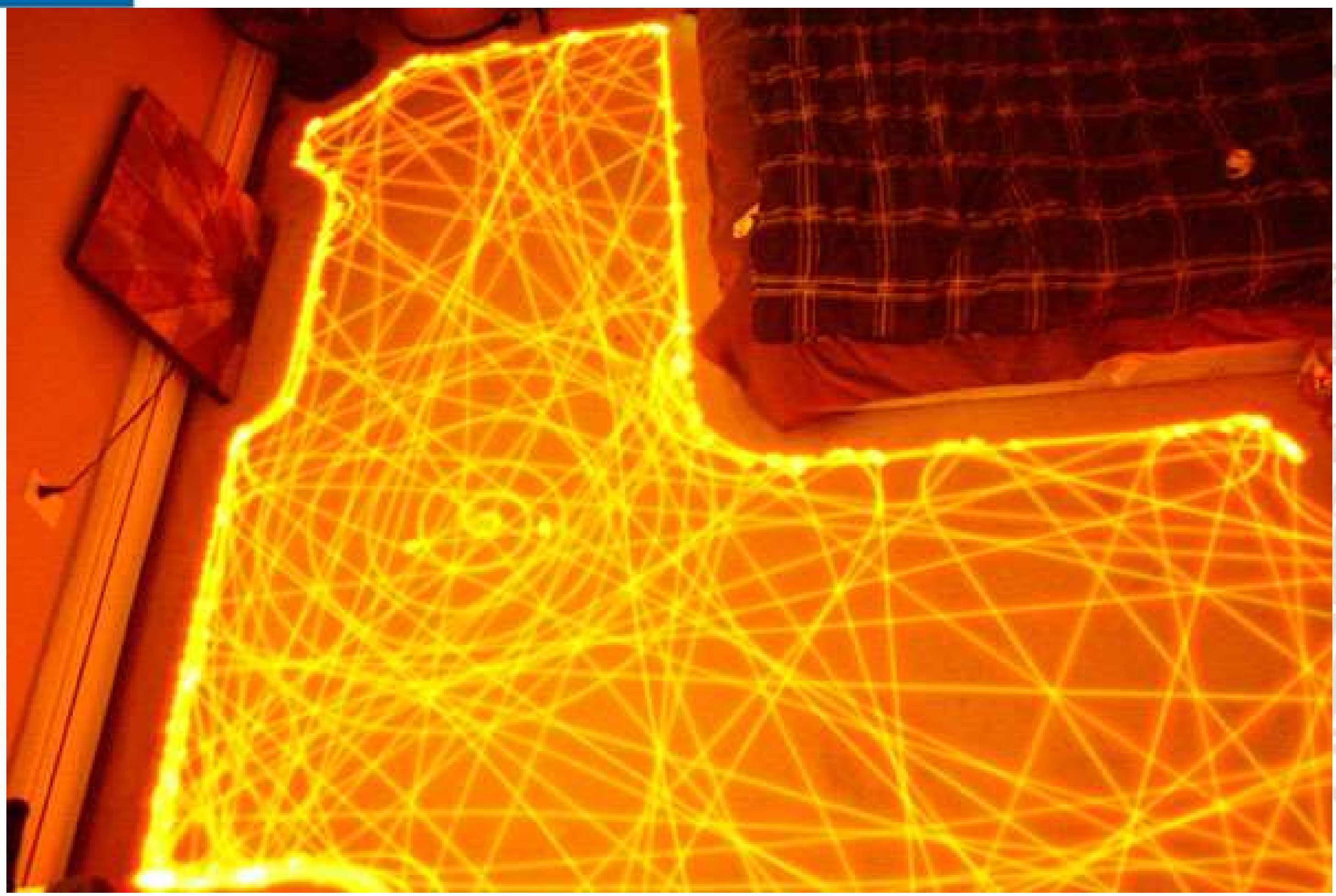
**Dirt Detection** (selected models)**:** When Roomba senses dirt, the blue Dirt Detect™ light is lit and Roomba cleans more intensely in that area.



THE BIOROBOTICS INSTITUTE

# iRobot Roomba
# Example of operation

Scuola Superiore Sant'Anna



https://www.youtube.com/watch?v=uCWeG3p5KJA