Distributed Intelligent Systems – W3
An Introduction to Sensing, Action, and Control in Mobile Robotics

Outline

- General concepts
  - Autonomy
  - Perception-to-action loop
  - Sensing, actuating, computing
- Education tools of the course
  - e-puck
  - Webots
- Main example of reactive control architectures
  - Proximal architectures
  - Distal architectures

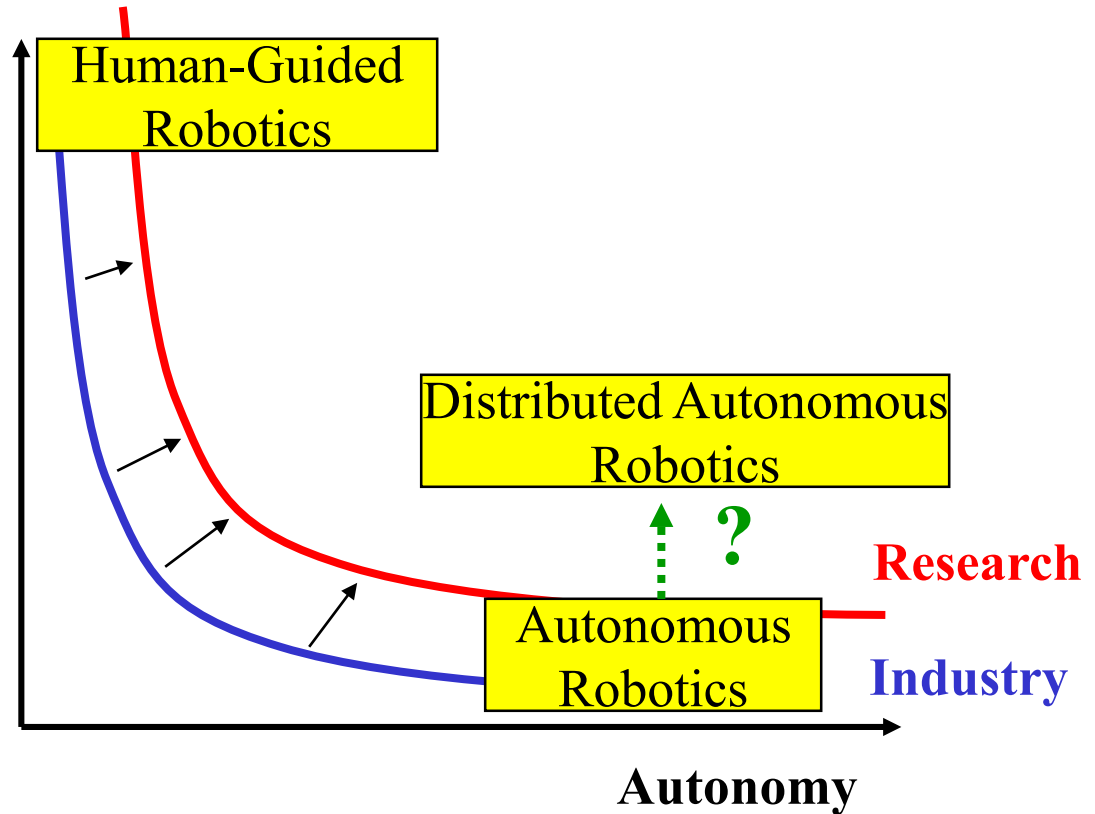# General Concepts and Principles for Mobile Robotics

Autonomy

- Different levels/degrees of autonomy
  - Energetic level
  - Sensory, motor, and computational level
  - Decisional level

- Needed degree of autonomy depends on task/environment in which the unit has to operate

- Environmental unpredictability is crucial: robot manipulator vs. mobile robot vs. sensor node

Autonomy – Mobile Robotics

Task Complexity

Human-Guided Robotics

State of the Art in Mobile Robotics

Distributed Autonomous Robotics

?

Research

Autonomous Robotics

Industry

Autonomy

Perception-to-Action Loop

- Reactive (e.g., nonlinear transform, single loop)
- Reactive + memory (e.g. filter, state variable, multi-loops)
- Deliberative (e.g. planning, multi-loops)

• sensors

• actuators

Perception

Computation

Action
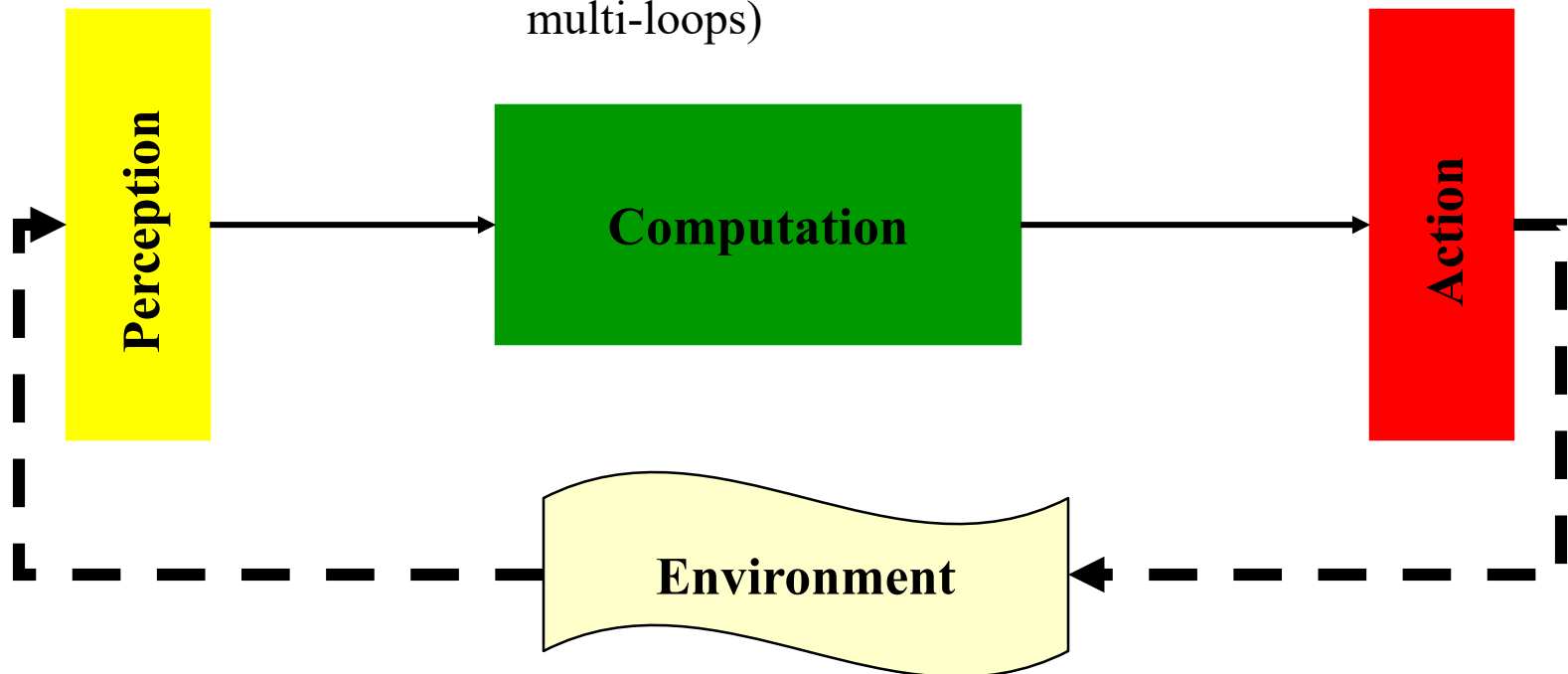
Environment

Sensors

- Proprioceptive ("body") vs. exteroceptive ("environment")

  – *Ex. proprioceptive*: motor speed/robot arm joint angle, battery voltage

  – *Ex. exteroceptive:* distance measurement, light intensity, sound amplitude

- Passive ("measure ambient energy") vs. active ("emit energy in the environment and measure the environmental reaction")

  – *Ex. passive*: temperature probes, microphones, cameras

  – *Ex. active*: laser rangefinder, IR proximity sensors, ultrasound sonars

Classification of Typical Sensors

General classification (typical use)	Sensor Sensor System	PC or EC	A or P
Tactile sensors (detection of physical contact or closeness; security switches)	Contact switches, bumpers	EC	P
	Optical barriers	EC	A
	Noncontact proximity sensors	EC	A
Wheel/motor sensors (wheel/motor speed and position)	Brush encoders	PC	P
	Potentiometers	PC	P
	Synchros, resolvers	PC	A
	Optical encoders	PC	A
	Magnetic encoders	PC	A
	Inductive encoders	PC	A
	Capacitive encoders	PC	A
Heading sensors (orientation of the robot in relation to a fixed reference frame)	Compass	EC	P
	Gyroscopes	PC	P
	Inclinometers	EC	A/P

A, active; P, passive; P/A, passive/active; PC, proprioceptive; EC, exteroceptive.

[From *Introduction to Autonomous Mobile Robots*, Siegwart R. and Nourbakhsh I. R.]

Classification of Typical Sensors

General classification (typical use)	Sensor Sensor System	PC or EC	A or P
Ground-based beacons (localization in a fixed reference frame)	GPS	EC	A
	Active optical or RF beacons	EC	A
	Active ultrasonic beacons	EC	A
	Reflective beacons	EC	A
Active ranging (reflectivity, time-of-flight, and geometric triangulation)	Reflectivity sensors	EC	A
	Ultrasonic sensor	EC	A
	Laser rangefinder	EC	A
	Optical triangulation (1D)	EC	A
	Structured light (2D)	EC	A
Motion/speed sensors (speed relative to fixed or moving objects)	Doppler radar	EC	A
	Doppler sound	EC	A
Vision-based sensors (visual ranging, whole-image analysis, segmentation, object recognition)	CCD/CMOS camera(s)	EC	P
	Visual ranging packages		
	Object tracking packages		

[From *Introduction to Autonomous Mobile Robots*, Siegwart R. and Nourbakhsh I. R.]

Action - Actuators

- **For different purposes**: locomotion, control a part of the body (e.g., arm), heating, sound producing, etc.

- **Examples** of electrical-to-mechanical actuators: DC motors, stepper motors, servos, loudspeakers, etc.

Computation

- Usually microcontroller-based; extra memory capabilities can be added and multiple micro-controllers are becoming standard

- ADC: Analog-to-Digital Conversion (continuous amplitude and time converted in discrete amplitude and time)

- DAC: Digital-to-Analog Conversion (discrete amplitude and time converted in continuous amplitude and time)

- Different types of control architectures: e.g., reactive ('reflex-based") vs. deliberative ("planning")

Sensor Performance

General Sensor Performance

– Range

  • Upper limit

– Dynamic range

  • ratio between lower and upper limits, usually in decibels (dB for power and amplitude)

  • e.g. voltage measurement from 1 mV to 20 V

$$20 \cdot \log\left[\frac{20}{0.001}\right] = 86 dB$$

Note: similar to the acoustic amplitude

  • e.g. power measurement from 1 mW to 20 W

$$10 \cdot \log\left[\frac{20}{0.001}\right] = 43 dB \qquad P = U \cdot I = \frac{1}{R}U^2$$

Note: 10 instead of 20 because power involves a squared amplitude!!

[Adapted from *Introduction to Autonomous Mobile Robots*, Siegwart R. and Nourbakhsh I. R.]

General Sensor Performance

– Resolution

- minimum difference between two values
- usually: lower limit of dynamic range = resolution
- for digital sensors it is usually the A/D resolution.
  - e.g. 8 bit A/D with upper range of 5V: resolution = 5 / 255

– Linearity

- variation of output signal as function of the input signal
- linearity is less important when signal is treated with a digital device (e.g., microcontroller, computer)

$$x \rightarrow f(x)$$
$$y \rightarrow f(y)$$
$$\alpha \cdot x + \beta \cdot y \rightarrow f(\alpha \cdot x + \beta \cdot y) \overset{?}{=} \alpha \cdot f(x) + \beta \cdot f(y)$$

[From *Introduction to Autonomous Mobile Robots*, Siegwart R. and Nourbakhsh I. R.]

General Sensor Performance

– Bandwidth or Frequency

- the speed with which a sensor can provide a stream of readings

- usually there is an upper limit depending on the sensor and the sampling rate

- lower limit is also possible, e.g. acceleration sensor

- frequency response: phase (delay, lag) of the signal and amplitude might be influenced

[Adapted from *Introduction to Autonomous Mobile Robots*, Siegwart R. and Nourbakhsh I. R.]

In Situ Sensor Performance

Characteristics that are especially relevant for real world environments

- Sensitivity
  - ratio of output change to input change
  - however, in real world environment, the sensor has very often high sensitivity to other environmental changes, e.g. illumination

- Cross-sensitivity (and cross-talk)
  - sensitivity to other environmental parameters
  - influence of other active sensors

- Error / Accuracy
  - difference between the sensor's output and the true value

error

$$\left(accuracy = 1 - \frac{|m - v|}{v} \right)$$

m = measured value
v = true value

[Adapted from *Introduction to Autonomous Mobile Robots*, Siegwart R. and Nourbakhsh I. R.]

In Situ Sensor Performance

Characteristics that are especially relevant for real world environments

- Systematic error -> deterministic errors
  - caused by factors that can (in theory) be modeled -> prediction
  - e.g. calibration of a laser sensor or of the distortion cause by the optic of a camera

- Random error -> non-deterministic
  - no deterministic prediction possible
  - however, they can be described probabilistically
  - e.g. gaussian noise on a distance sensor, black level noise of camera

- Precision (different from accuracy!)
  - *reproducibility* of sensor results

$$precision = \frac{range}{\sigma}$$

σ = *standard dev of the sensor noise*

[From *Introduction to Autonomous Mobile Robots*, Siegwart R. and Nourbakhsh I. R.]

Educational Tools for This Course

The e-puck Mobile Robot

http://www.e-puck.org/

Main features

- Cylindrical, Ø 70mm
- dsPIC processor
- Two stepper motors
- Ring of LEDs
- Many sensors:
  - ✓ Camera
  - ✓ Sound
  - ✓ IR proximity
  - ✓ 3D accelerometer
- Li-ion accumulator
- Bluetooth wireless communication
- Open hardware (and software)
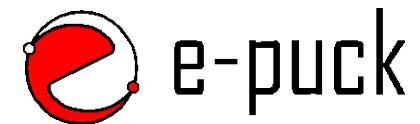
e-puck

e-puck Block Schema

Actuators?

Sensors?

Computation?

Communication?
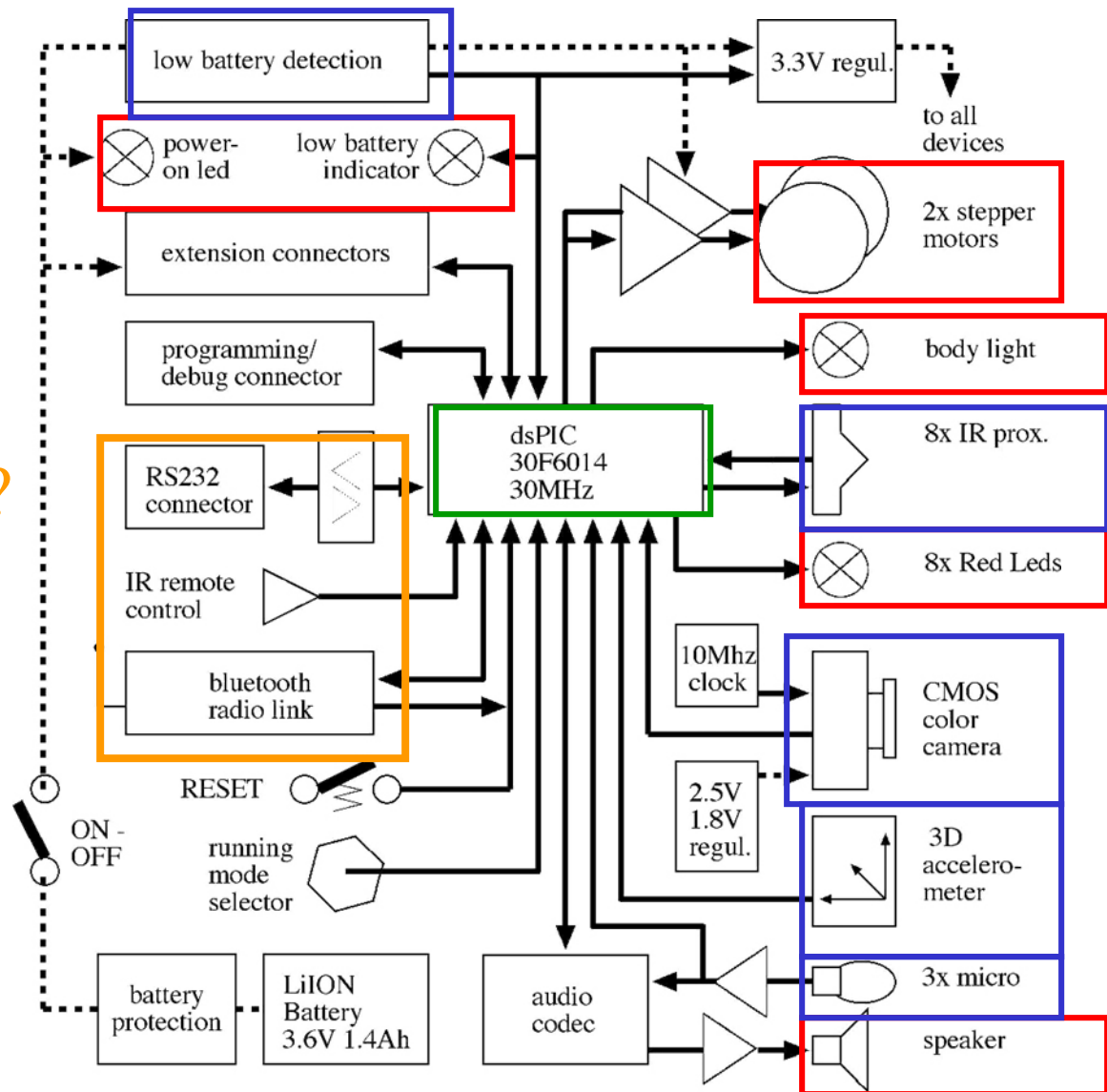
Simulation: Why?

- Hardware prototyping is time-consuming and in general more expensive

- Flexibility in the experimental setup

- Easier for monitoring experiments (and evaluating specific metrics)

- Evaluation of algorithms in settings difficult (e.g., very large number of robots) or even impossible (e.g., noise-free sensors) to reproduce in reality

- Predictive value: asking questions in simulation before building hardware

Real and Simulated e-puck

Real e-puck
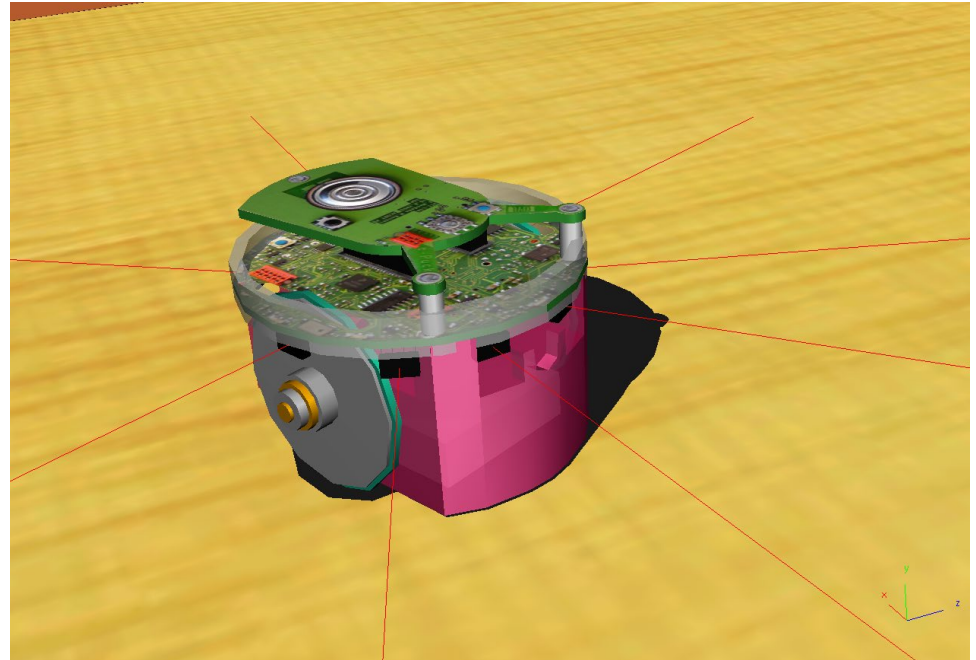
Realistically simulated e-puck (Webots)
- intra robot details: discrete sensors, actuators, transceivers, etc.
- noise, nonlinearities of S&A reproduced

Webots: High-Fidelity Simulator
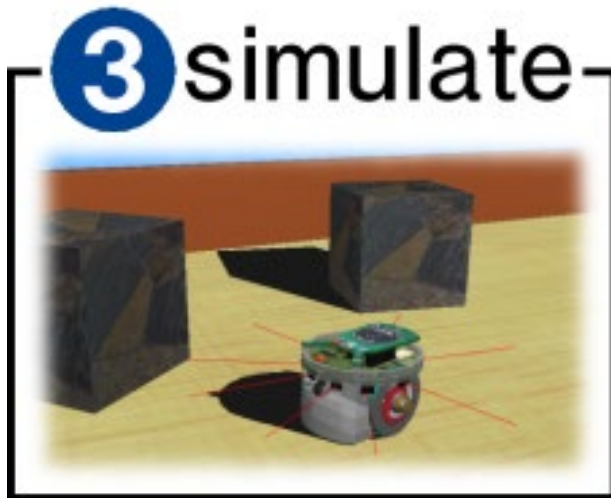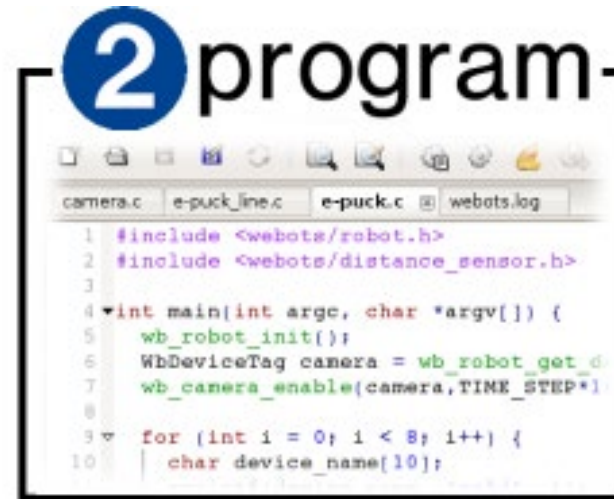
In this course, we will focus on steps **2** and **3** only.

Webots Features

- As of Jan 2019 open source

- Faithful physics-based robotics simulator

- Tuneable trade-off between faithfulness and computational cost through kinematic (e.g., speed, position) vs. dynamic (e.g., forces, friction, mass) modes

- Fast prototyping; can be interfaced with ROS

- Can *usually* run faster than real-time

- Available sensors: distance sensors, light sensors, cameras, accelerometers, touch sensors, position sensors, GPSs, receivers, force sensors, etc.

- Available actuators: linear and rotational motors, grippers, LEDs, emitters, etc.

Webots Principles
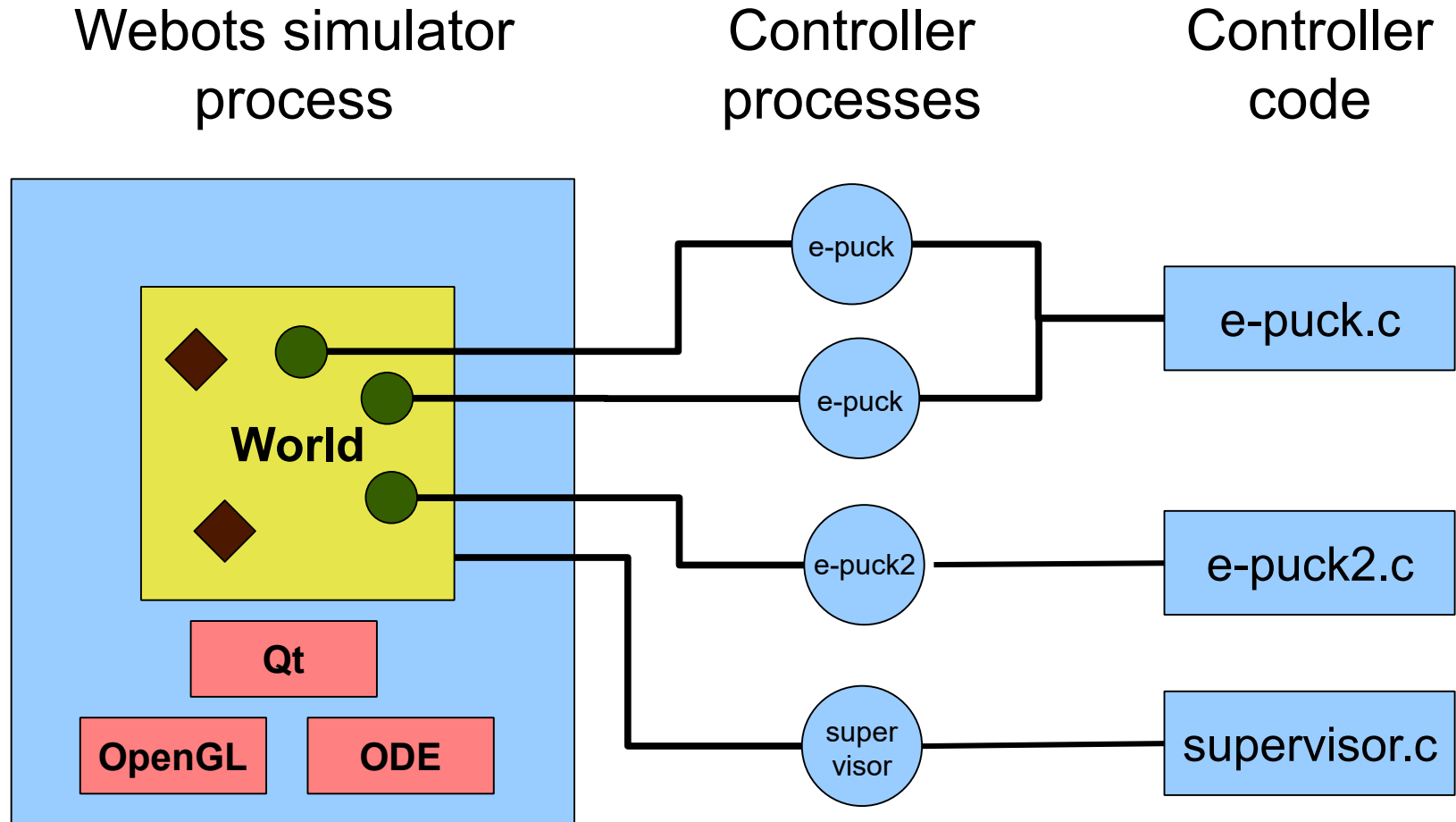
The more robots, the slower the simulation!

Webots GUI

scene tree

world view

editor

simulation speed
(w.r.t. real-time)

console

26
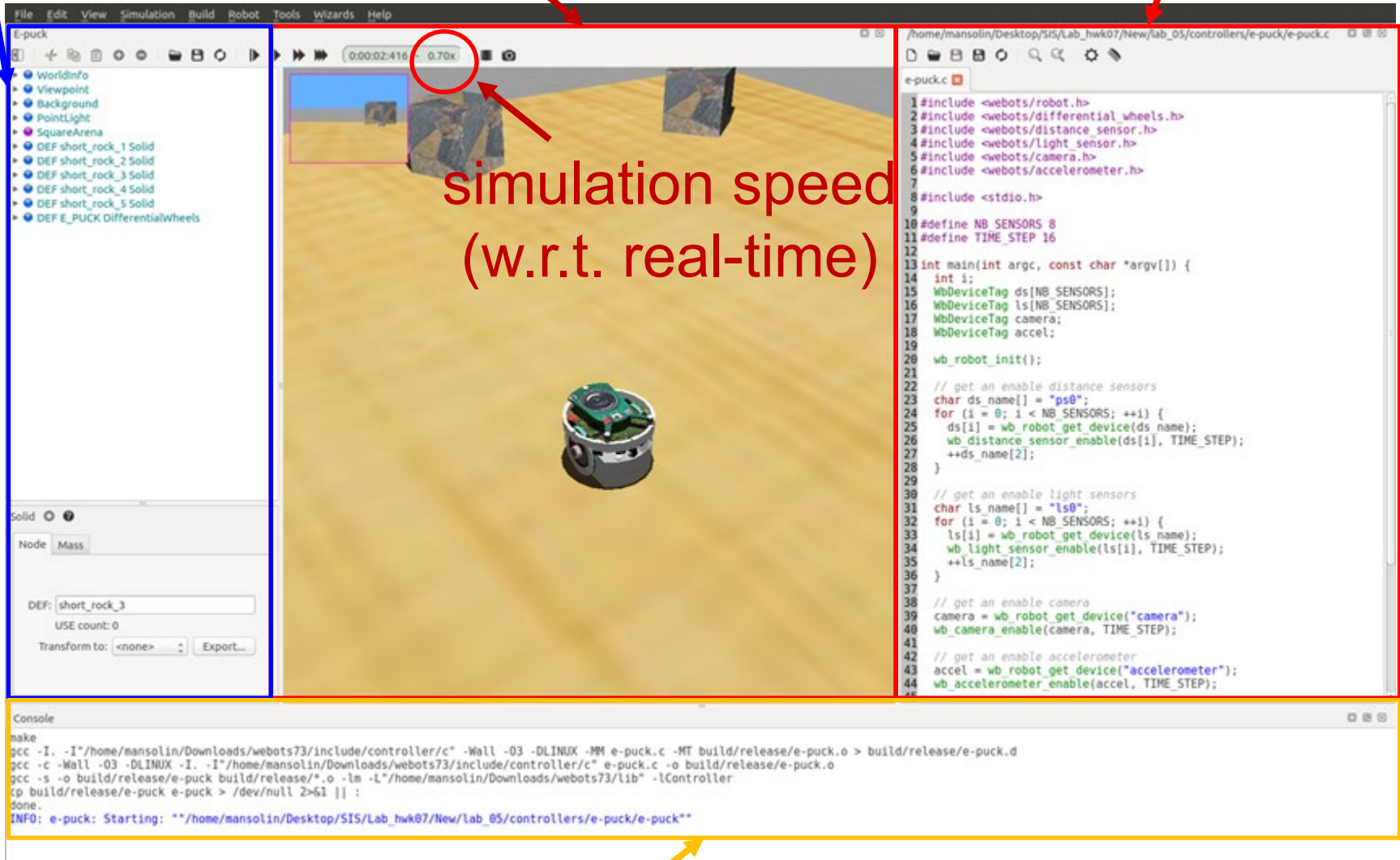
Examples of Reactive Control Architectures

Reactive Architectures: Proximal vs. Distal in Theory

- Proximal:
  - close to sensor and actuators
  - very simple linear/nonlinear operators on crude data
  - high flexibility in shaping the behavior
  - Difficult to engineer in a "human-guided" way; machine-learning usually perform better

Reactive Architectures: Proximal vs. Distal in Theory

- Distal architectures
  - Further from sensor and actuators
  - Self-contained behavioral blocks
  - Less flexibility in shaping the behavior
  - Easier to engineer in a "human-guided" way the basic block (handcoding); more difficult to compose the blocks in the right way (e.g., sequence, parallel, …)

Reactive Architectures: Proximal vs. Distal in Practice

- A whole blend!

- Five "classical" examples of reactive control architecture for solving the same problem: obstacle avoidance.

- Two proximal: Braitenberg and Artificial Neural Network

- Three distal: rule-based and two behavior-based (Subsumption and Motor Schema)

Ex. 1: Braitenberg's Vehicles
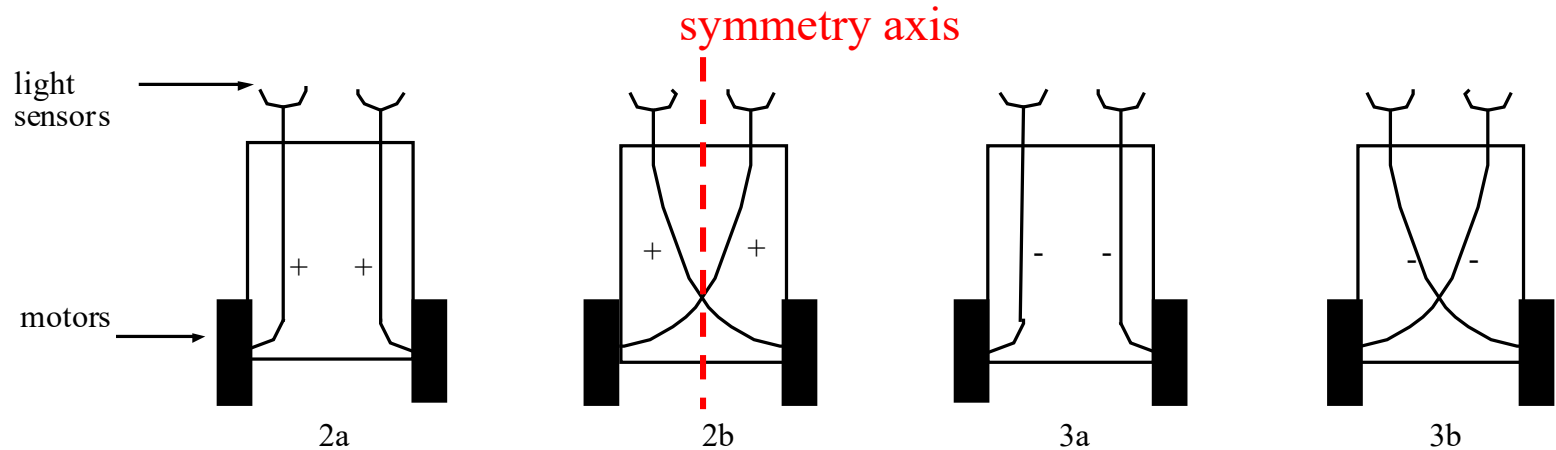
- Work on the difference (gradient) between sensors
- + excitation, - inhibition; linear controller (output = signed coefficient * input)
- Symmetry axis along main axis of the vehicle (----)
- Originally omni-directional sensors but work even better with directional sensors
- Originally: light sensors; works perfectly also with proximity sensors (3c?)
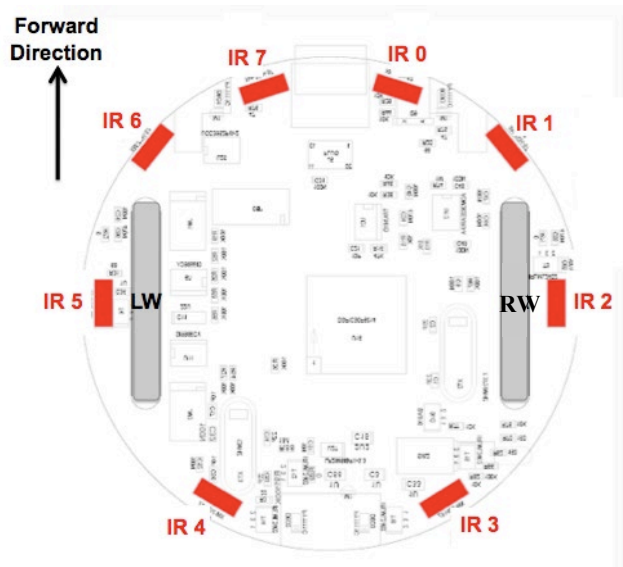
Examples of Braitenberg's Vehicles
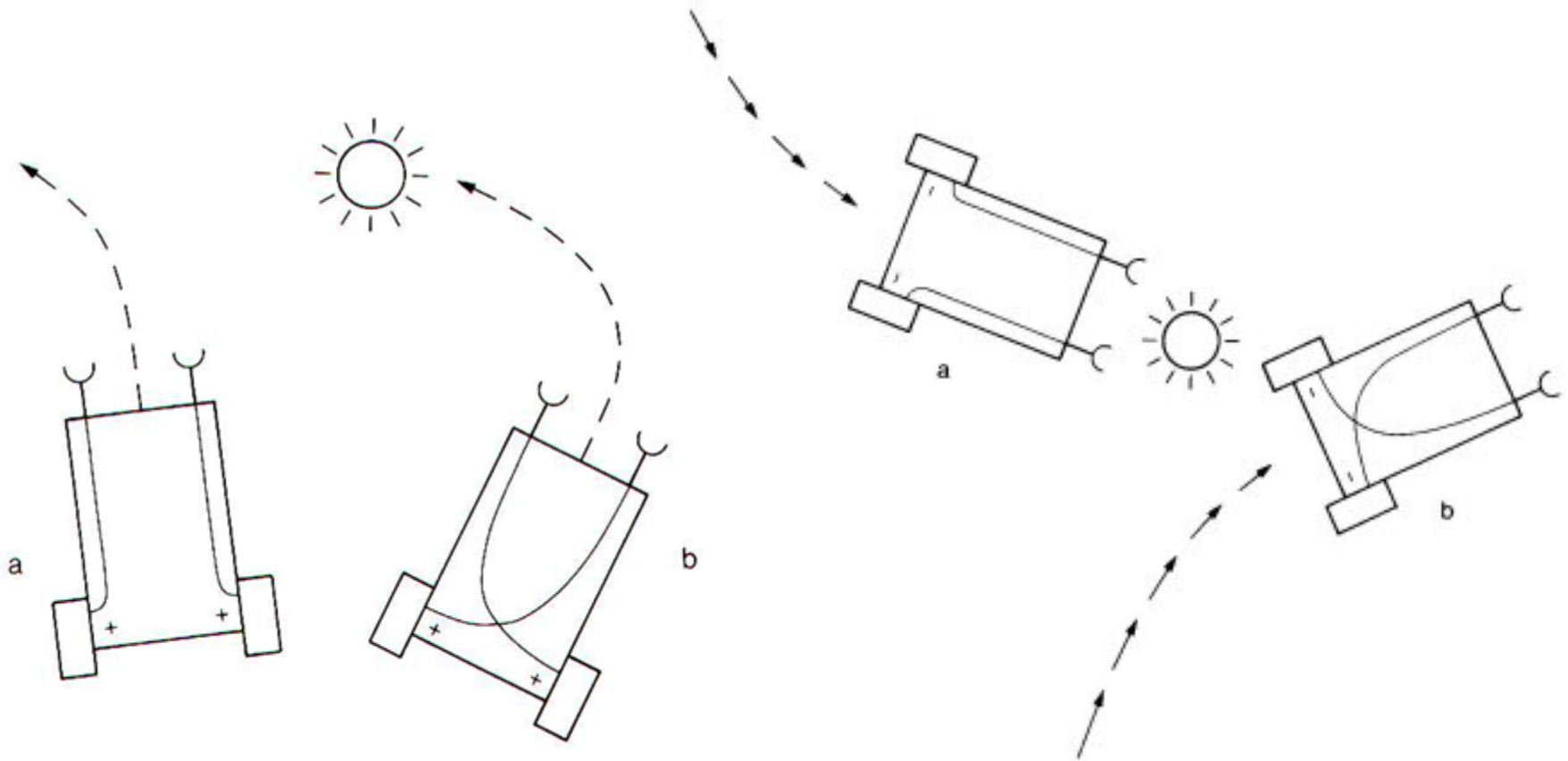
Excitatory connections   Inhibitory connections
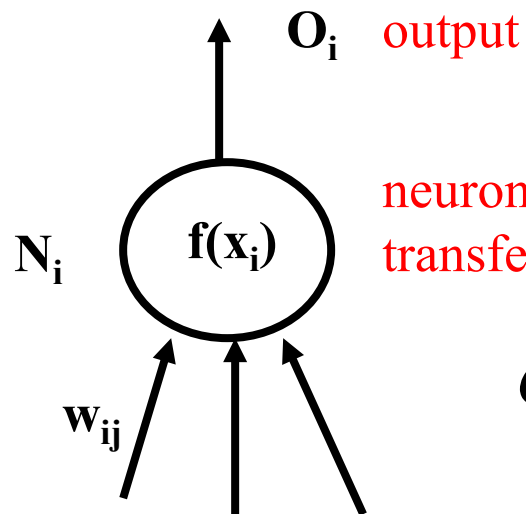
Braitenberg Applied to e-puck

- 2 actuators

- 8 proximity sensors

- Motor speed is a linear combination:

$$\begin{bmatrix} v_L \\ v_R \end{bmatrix} = \begin{bmatrix} \alpha_{L0} & \alpha_{L1} & \cdots & \alpha_{L7} \\ \alpha_{R0} & \alpha_{R1} & \cdots & \alpha_{R7} \end{bmatrix} \cdot \begin{bmatrix} d_{IR0} \\ \vdots \\ d_{IR7} \end{bmatrix} + \begin{bmatrix} v_{L0} \\ v_{R0} \end{bmatrix}$$

bias
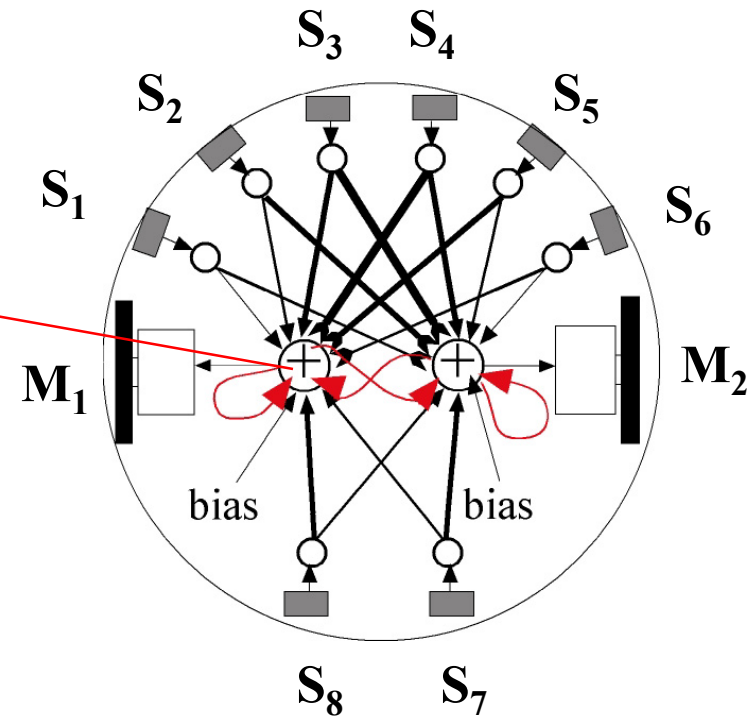
Ex. 2: Artificial Neural Network

$\mathbf{O_i}$ output

$\mathbf{N_i}$ $\mathbf{f(x_i)}$

neuron N with sigmoid transfer function f(x)

$\mathbf{w_{ij}}$

synaptic weight

$\mathbf{I_j}$ input
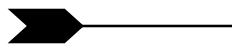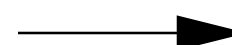
$$O_i = f(x_i)$$

$$f(x) = \frac{2}{1 + e^{-x}} - 1$$

$$x_i = \sum_{j=1}^{m} w_{ij} I_j + I_0$$

S_2 S_3 S_4 S_5

S_1 S_6

M_1 bias bias M_2

S_8 S_7

inhibitory conn.
excitatory conn.

Ex. 3: Rule-Based

Rule 1:
if (proximity sensors on the left active) **then**
  turn right

Rule 2:
**if** (proximity sensors on the right active) **then**
  turn left

Rule 3:
if (no proximity sensors active) **then**
  move forwards

Subsumption Architecture

- Rodney Brooks 1986, MIT

- Precursors: Braitenberg (1984), Walter (1953)

- Behavioral modules (basic behaviors) represented by Augmented Finite State machines (AFSM)

- Response encoding: predominantly discrete (rule based)

- Behavioral coordination method: competitive (priority-based arbitration via inhibition and suppression)
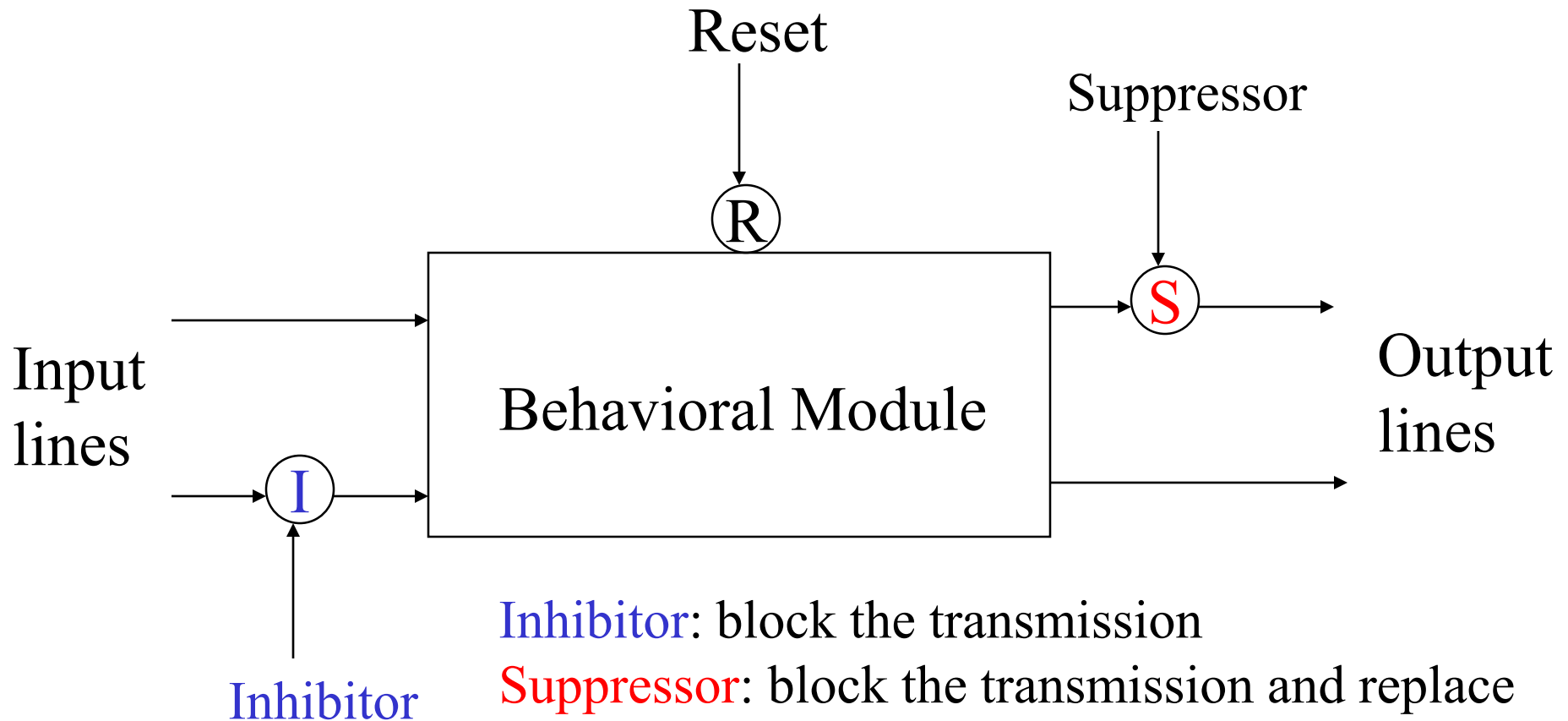
Subsumption Architecture
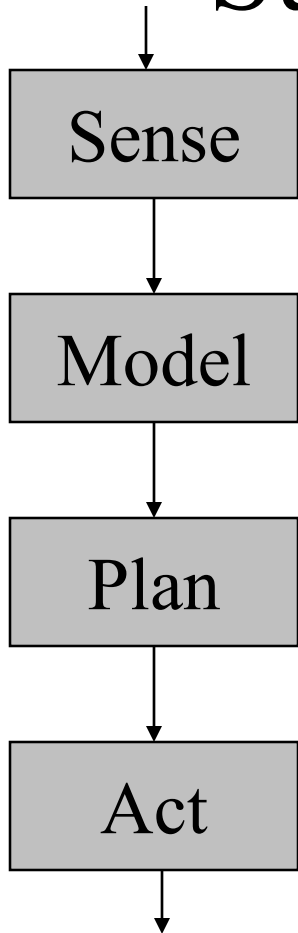
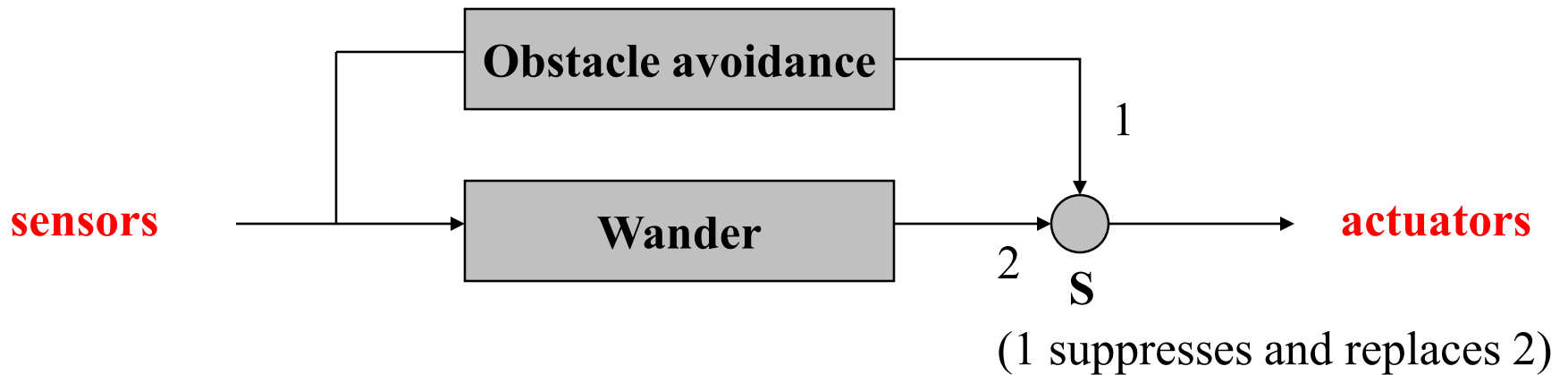**Classical paradigm (serial); emphasis on deliberative control**

Subsumption (parallel); emphasis on reactive control

Subsumption Architecture: AFSM

Reset

Suppressor

Ⓡ

Ⓢ

Input lines

Ⓘ

Behavioral Module

Output lines

Inhibitor

Inhibitor: block the transmission
Suppressor: block the transmission and replace the signal with the suppressing message

Ex. 4: Behavior-Based with Subsumption

sensors → Obstacle avoidance / Wander → S → actuators

1

2

S

(1 suppresses and replaces 2)

Evaluation of Subsumption

\+ Support for parallelism: each behavioral layer can run independently and asynchronously (including different loop time)

\+ Fast execution time possible

\- Coordination mechanisms restrictive ("black or white")

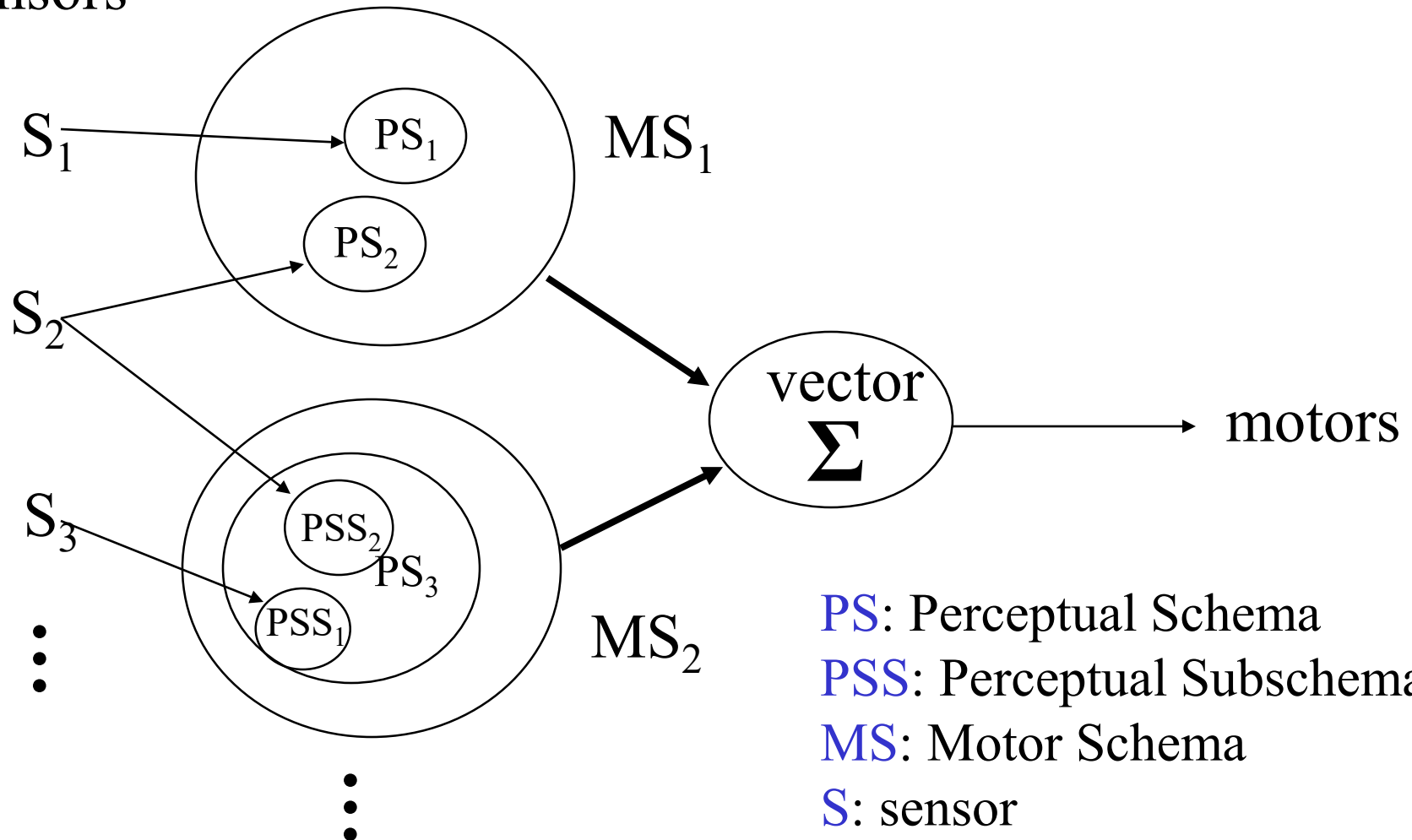\- Limited support for modularity (upper layers design cannot be independent from lower layers).
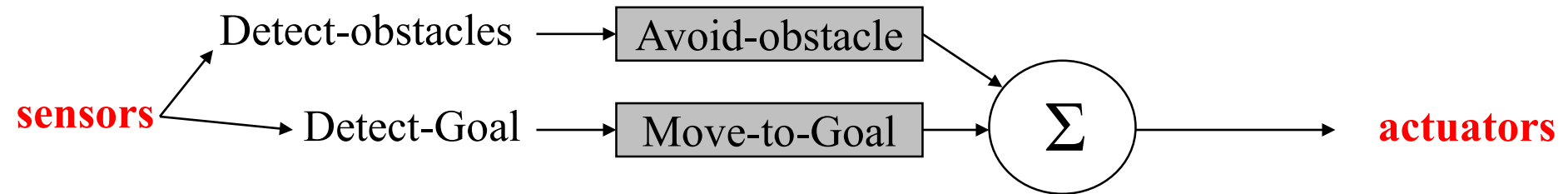
Motor Schemas

- Ronald Arkin 1987, Georgia Tech

- Precursors: Arbib (1981), Khatib (1985)

- Parametrized behavioral libraries (schemas)

- Response encoding: continuous using potential field analog

- Behavioral coordination method: cooperative via vector summation and normalization

Motor Schemas

sensors

S_1 → PS$_1$ MS$_1$

S_2 → PS$_2$

S_3 → PSS$_2$ PS$_3$
         PSS$_1$ MS$_2$

vector Σ → motors

PS: Perceptual Schema
PSS: Perceptual Subschema
MS: Motor Schema
S: sensor

Ex. 5: Behavior-Based with Motor Schemas

Visualization of Vector field for Ex. 5

Avoid-static-obstacle

$$V_{magnitude} = \begin{cases} 0 & for & d > S \\ \dfrac{S-d}{S-R}G & for & R < d \le S \\ \infty & for & d \le R \end{cases}$$
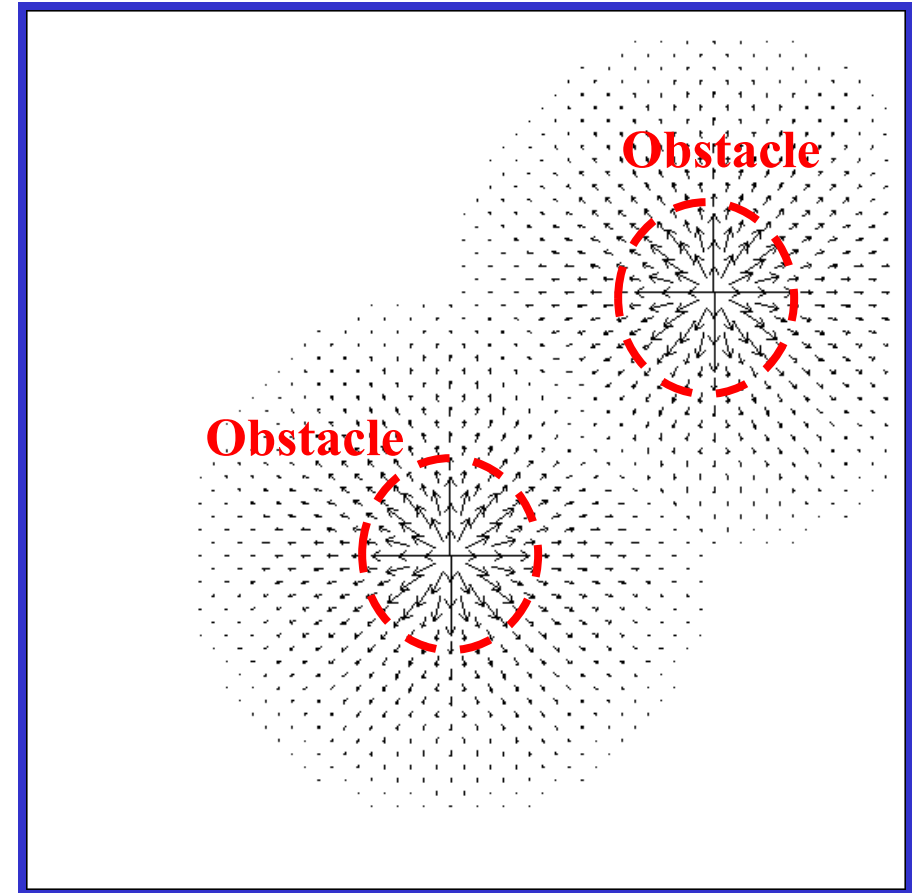
S = obstacle's sphere of influence

R = radius of the obstacle

G = gain

d = distance robot to obstacle's center

$V_{direction}$ = radially along a line
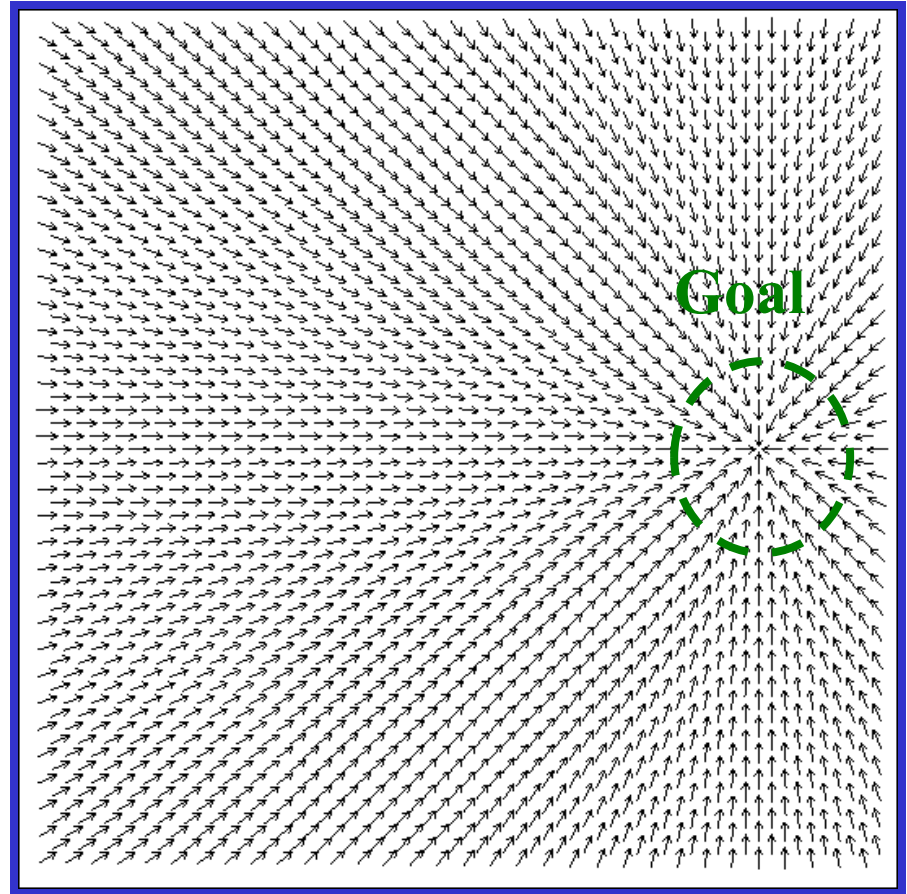between robot and
obst. center, directed
away from the obstacle

Obstacle

Obstacle

Visualization of Vector field for Ex. 5

Move-to-goal (ballistic)

Output = vector = (r, φ)
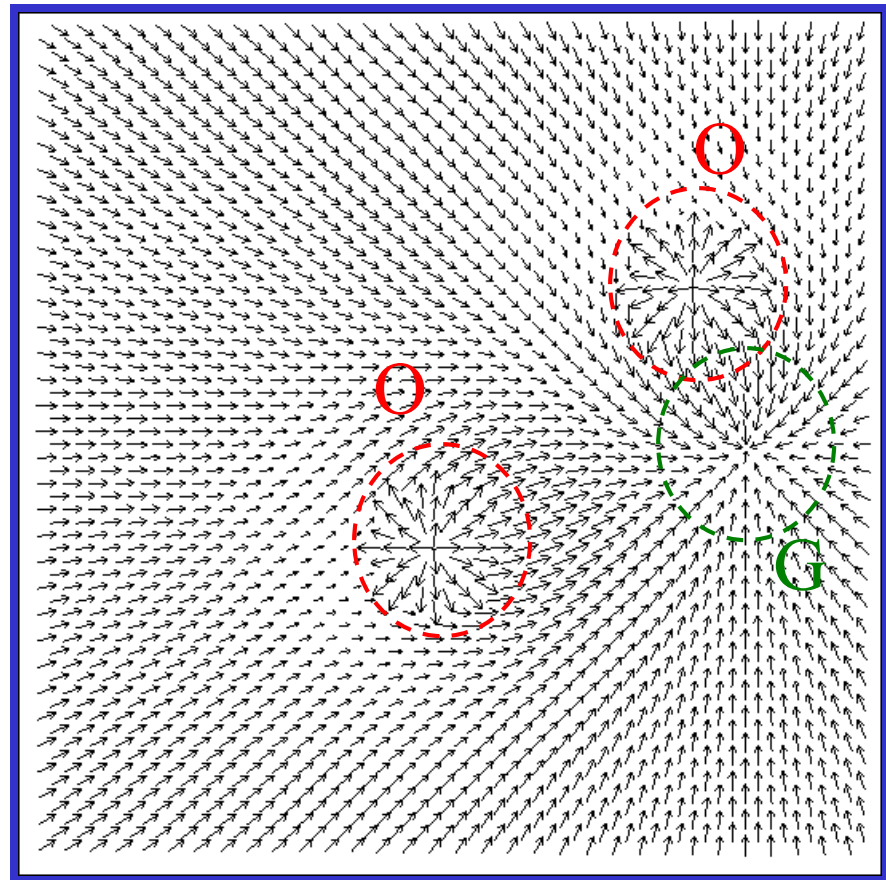(magnitude, direction)

$V_{magnitude}$ = fixed gain value

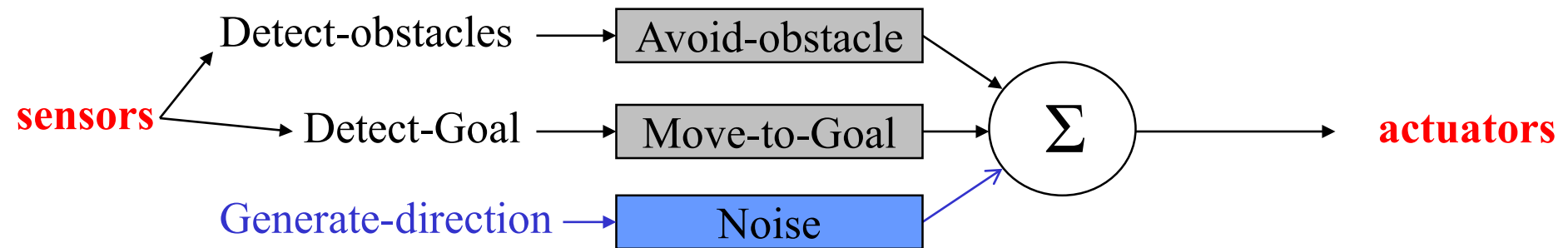$V_{direction}$ = towards perceived
        goal

Goal

Visualization of Vector field for Ex. 5

Move-to-goal + avoid obstacle

Linear combination
(weigthed sum)

Ex. 5: Behavior-Based with Motor Schemas

Detect-obstacles $\longrightarrow$ Avoid-obstacle

sensors

Detect-Goal $\longrightarrow$ Move-to-Goal $\longrightarrow$ Σ $\longrightarrow$ **actuators**

Generate-direction $\longrightarrow$ Noise

Adding noise is a simple solution for avoiding to get stuck in local minima using arbitrary vector fields

Alternative more complex approach: use harmonic potential functions (explicitly designed for not generating local minima)

Evaluation of Motor Schemas

+ Support for parallelism: motor schemas are naturally parallelizable

+ Fine-tuned behavioral blending possible

- Robustness -> well-known problems of potential field approach -> extra introduction of noise or more complex functions

- Slow and computationally expensive sometimes

Evaluation of both Architectures in Practice

- In practice (my expertise) you tend to mix both and even more …

- The way to combine basic behavior (cooperative and/or competitive) depends from how you developed the basic behaviors (or motor schemas), reaction time required, on-board computational capabilities, …

Conclusion

Take Home Messages

- Perception-to-action loop is key in robotics, several sensor and actuator modalities
- Key categories for sensor classification are exteroceptive vs. proprioceptive and active vs. passive
- Experimental work can be carried out with real and realistically simulated robots
- A given behavior can be obtained with different control architectures
- Control architectures can be roughly classified in proximal and distal architectures
- Braitenberg vehicles and artificial neural networks are two typical examples of proximal architectures, motor schemas and subsumption are typical example of distal architectures

Additional Literature – Week 3

Books

- Braitenberg V., "Vehicles: Experiments in Synthetic Psychology", MIT Press, 1986.
- Siegwart R., Nourbakhsh I. R., and Scaramuzza D., "Introduction to Autonomous Mobile Robots", 2nd edition, MIT Press, 2011.
- Arkin R. C., "Behavior-Based Robotics". MIT Press, 1998.
- Everett, H. R., "Sensors for Mobile Robots, Theory and Application", A. K. Peters, Ltd., 1995