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

• e-puck
  – Basic features
  – HW architecture
  – Simulation

• 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

State of the Art in Mobile Robotics

Task Complexity

Human-Guided Robotics

Distributed Autonomous Robotics

Autonomous Robotics

Research

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)

Environment

Perception

Computation

Action

- sensors

- actuators
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

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

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

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

[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

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

Note: similar to the acoustic amplitude

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

\[ 10 \cdot \log \left[ \frac{20}{0.001} \right] = 43dB \]

\[ P = U \cdot I = \frac{1}{R} U^2 \]

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)

\[ x \rightarrow f(x) \]
\[ y \rightarrow f(y) \]
\[ \alpha \cdot x + \beta \cdot y \rightarrow f(\alpha \cdot x + \beta \cdot y) = \alpha \cdot f(x) + \beta \cdot f(y) \]

[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

\[
\text{accuracy} = 1 - \frac{|m - v|}{v} \quad m = \text{measured value} \quad v = \text{true value}
\]

[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
  \[ \text{precision} = \frac{\text{range}}{\sigma} \]
  \[ \sigma = \text{standard dev of the sensor noise} \]

[From Introduction to Autonomous Mobile Robots, Siegwart R. and Nourbakhsh I. R.]
e-puck: An Educational Robotic Tool
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 Overview

- IR receiver (remote control)
- Accelerometer
- Programming and debug connector
- ON-OFF
- microphones
- Wheels with stepper motor
- RS232
- Reset
- Mode selector
- Speaker
- Ring of LEDs
- IR proximity sensors
- CMOS camera
- Li-Ion accumulator
e-puck Mechanical Structure
PIC/dsPIC Family

from www.microchip.com

Microcontroller on the e-puck
# dsPIC Characteristics

## TABLE 1-1: dsPIC30F GENERAL PURPOSE FAMILY VARIANTS

| Device             | Pins | Program Memory | SRAM Bytes | EEPROM Bytes | Timer 16-bit | Input Capture | Output Compare | ADC 12-bit | UART | SPI | PC | CAN | IO Pins (Max., | Packages (o) |
|--------------------|------|----------------|------------|--------------|--------------|---------------|----------------|------------|------|-----|----|-----| Max.)       |              |
| dsPIC30F3014       | 40/44| 24K            | 8K         | 2048         | 1024         | 3             | 2              | 2          |      |     |    |     |              | 30           | PG, PT        |
| dsPIC30F4013       | 40/44| 48K            | 16K        | 2048         | 1024         | 5             | 4              | 4          | AC'97| I2S |    |     |              | 30           | PG, PT        |
| dsPIC30F5011       | 64   | 66K            | 22K        | 4096         | 1024         | 5             | 8              | 8          | AC'97| I2S |    |     |              | 52           | PT            |
| dsPIC30F8011(3)    |      |                |            |              |              |               |                |            |      |     |    |     |              |              |               |
| dsPIC30F8011A      | 64   | 132K           | 44K        | 6144         | 2048         | 5             | 8              | 8          |      | 16  |    |    | 52            |               | PF, PT        |
| dsPIC30F6012(3)    |      |                |            |              |              |               |                |            |      |     |    |     |              |              |               |
| dsPIC30F6012A      | 64   | 144K           | 48K        | 8192         | 4096         | 5             | 8              | 8          | AC'97| I2S |    |     |              | 52           | PF, PT        |
| dsPIC30F5013       | 80   | 66K            | 22K        | 4096         | 1024         | 5             | 8              | 8          | AC'97| I2S |    |     |              | 68           | PT            |
| dsPIC30F6013(3)    |      |                |            |              |              |               |                |            |      |     |    |     |              |              |               |
| dsPIC30F6013A      | 80   | 132K           | 44K        | 6144         | 2048         | 5             | 8              | 8          |      | 16  |    |    | 68           | PF, PT        |
| dsPIC30F6011(3)    |      |                |            |              |              |               |                |            |      |     |    |     |              |              |               |
| dsPIC30F6011A      | 80   | 144K           | 48K        | 8192         | 4096         | 5             | 8              | 8          | AC'97| I2S |    |     |              | 68           | PF, PT        |

**Notes:**
- e-puck microcontroller

e-puck Block Schema

Actuators?

Sensors?

Computation?

Communication?
Light and Proximity Sensors

- Light and proximity sensors incorporated in the same electronic component.
- Khepera II has essentially the same components of e-pucks for this functionality.
- e-puck sensor numbering slightly different from Khepera II (e.g. IR7 and IR0 correspond to Sensor 2 and 3), but same layout.

Khepera II in front of a 50 W bulb

Khepera II in front of white paper
Low to medium sampling frequency application

- Typically a function of the application and of the accelerometer characteristics
- Sampling of the continuous time analog accelerometer (3 axes) using the integrated A/D converter

Table 2. Operating Characteristics

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Symbol</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supply Voltage(3)</td>
<td>VDD</td>
<td>2.2</td>
<td>3.3</td>
<td>3.6</td>
<td>V</td>
</tr>
<tr>
<td>Supply Current</td>
<td>IDD</td>
<td>—</td>
<td>500</td>
<td>800</td>
<td>µA</td>
</tr>
<tr>
<td>Supply Current at Sleep Mode(4)</td>
<td>IDD</td>
<td>—</td>
<td>3</td>
<td>10</td>
<td>µA</td>
</tr>
<tr>
<td>Operating Temperature Range</td>
<td>T_A</td>
<td>-20</td>
<td>—</td>
<td>85</td>
<td>°C</td>
</tr>
<tr>
<td>Acceleration Range, X-Axis, Y-Axis, Z-Axis</td>
<td>gRS</td>
<td>—</td>
<td>±1.5</td>
<td>—</td>
<td>g</td>
</tr>
<tr>
<td>g-Select1 &amp; 2: 00</td>
<td>gRS</td>
<td>—</td>
<td>±2.0</td>
<td>—</td>
<td>g</td>
</tr>
<tr>
<td>g-Select1 &amp; 2: 10</td>
<td>gRS</td>
<td>—</td>
<td>±4.0</td>
<td>—</td>
<td>g</td>
</tr>
<tr>
<td>g-Select1 &amp; 2: 01</td>
<td>gRS</td>
<td>—</td>
<td>±9.0</td>
<td>—</td>
<td>g</td>
</tr>
<tr>
<td>Output Signal</td>
<td>V_OFF</td>
<td>1.485</td>
<td>1.65</td>
<td>1.815</td>
<td>V</td>
</tr>
<tr>
<td>Zero g</td>
<td>V_OFF, T_A</td>
<td>—</td>
<td>±2</td>
<td>—</td>
<td>mg/°C</td>
</tr>
<tr>
<td>Sensitivity (T_A = 25°C, V_DD = 3.3 V)(5)</td>
<td>S_1.5g</td>
<td>740</td>
<td>800</td>
<td>800</td>
<td>mV/g</td>
</tr>
<tr>
<td>2g</td>
<td>S_2g</td>
<td>555</td>
<td>600</td>
<td>645</td>
<td>mV/g</td>
</tr>
<tr>
<td>4g</td>
<td>S_4g</td>
<td>277.5</td>
<td>300</td>
<td>322.5</td>
<td>mV/g</td>
</tr>
<tr>
<td>6g</td>
<td>S_6g</td>
<td>185</td>
<td>200</td>
<td>215</td>
<td>mV/g</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>S_TA</td>
<td>—</td>
<td>±3</td>
<td>—</td>
<td>%/°C</td>
</tr>
<tr>
<td>Bandwidth Response</td>
<td>f3dB</td>
<td>—</td>
<td>350</td>
<td>—</td>
<td>Hz</td>
</tr>
<tr>
<td>XY</td>
<td>f3dB</td>
<td>—</td>
<td>150</td>
<td>—</td>
<td>Hz</td>
</tr>
</tbody>
</table>

±1.5g - 6g Three Axis Low-g Micromachined Accelerometer

- MMA7260Q low cost capacitive micromachined accelerometer
- Features:
  - Selectable Sensitivity (1.5g, 2g, 4g, or 6g)
  - Low Current Consumption: 500 µA
  - Sleep Mode: 2 µA
  - Low Voltage Operation: 2.2 V to 3.6 V
  - ± 5 V ± 50 ppm x 14 DOF (3D)
  - ± 500 mV/mg @ 25°C
  - Fast Turn On Time
  - High Sensitivity (± 1.5g)
  - Integral Signal Conditioning with Low Pass Filter
  - Board Design, High Shock survivability
  - Pin-Free Terminations
  - Environmentally Preferred Package
  - Low Cost

Actual Fall Data (From 22 inch height, lap top)
Acoustic Sensors

Medium to high sampling frequency application

Example: acoustic source localization

- Robot dimension: 7.5 cm → microphone max inter-distance: 5.5 cm → speed of sound in air: 340 m/s → travel time microphone-to-microphone: 0 (orthogonal) to 161 μs (aligned).

- max DsPIC sampling frequency (1 channel): 200 KHz (see datasheet)

- 2 microphones: 2 ch., e.g., 85 kHz → 12 μs → 4 mm resolution but possible aliasing on a plane (dual localization)

- 3 microphones: 3 ch., e.g. 56 kHz → 18 μs → 6 mm resolution but no aliasing on a plane (unique localization)
Vision Sensor (Camera)

General requirements for embedded vision: handling of very large data flow (tens of Mbit/s)

Processing:
- Pixels H x V x RGB x fps
- $640 \times 480 \times 3 \times 30 = 27$Mbytes/second
- The dsPIC can execute max 15MIPS (millions of instructions/second)

Memory
- One image RGB (8,8,8 bits) of 640x480 uses 922kbytes
- Our dsPIC has 8kbytes of RAM (Random Access Memory), for variables
- Full image acquisition impossible

---

**Table 1-1: dsPIC30F GENERAL PURPOSE FAMILY VARIANTS**

<table>
<thead>
<tr>
<th>Device</th>
<th>Pins</th>
<th>Program Memory</th>
<th>RAM Memory</th>
<th>Timer-16-bit</th>
<th>Input Capture</th>
<th>Codec Interface</th>
<th>Analog 12-bit 200 kbps</th>
<th>UART</th>
<th>SPI™</th>
<th>I²C™</th>
<th>CAN</th>
<th>VO Pins (Min./Max.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>dsPIC30F3014</td>
<td>40/44</td>
<td>25K</td>
<td>8K</td>
<td>2048</td>
<td>1024</td>
<td>3</td>
<td>13 ch</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>30</td>
<td>PG, PT</td>
</tr>
<tr>
<td>dsPIC30F4013</td>
<td>40/44</td>
<td>48K</td>
<td>16K</td>
<td>2048</td>
<td>1024</td>
<td>5</td>
<td>13 ch</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>30</td>
<td>PG, PT</td>
</tr>
<tr>
<td>dsPIC30F5011</td>
<td>64</td>
<td>22K</td>
<td>8K</td>
<td>4096</td>
<td>1024</td>
<td>5</td>
<td>16 ch</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>52</td>
<td>PT</td>
</tr>
<tr>
<td>dsPIC30F6011(1)</td>
<td>64</td>
<td>22K</td>
<td>8K</td>
<td>6144</td>
<td>2048</td>
<td>5</td>
<td>16 ch</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>52</td>
<td>PT</td>
</tr>
<tr>
<td>dsPIC30F6011(2)</td>
<td>64</td>
<td>22K</td>
<td>8K</td>
<td>6144</td>
<td>2048</td>
<td>5</td>
<td>16 ch</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>52</td>
<td>PT</td>
</tr>
<tr>
<td>dsPIC30F6012A</td>
<td>64</td>
<td>22K</td>
<td>8K</td>
<td>6144</td>
<td>2048</td>
<td>5</td>
<td>16 ch</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>52</td>
<td>PT</td>
</tr>
<tr>
<td>dsPIC30F5015</td>
<td>60</td>
<td>22K</td>
<td>8K</td>
<td>4096</td>
<td>1024</td>
<td>5</td>
<td>16 ch</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>68</td>
<td>PT</td>
</tr>
<tr>
<td>dsPIC30F6013(1)</td>
<td>60</td>
<td>22K</td>
<td>8K</td>
<td>6144</td>
<td>2048</td>
<td>5</td>
<td>16 ch</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>68</td>
<td>PT</td>
</tr>
<tr>
<td>dsPIC30F6013(2)</td>
<td>60</td>
<td>22K</td>
<td>8K</td>
<td>6144</td>
<td>2048</td>
<td>5</td>
<td>16 ch</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>68</td>
<td>PT</td>
</tr>
<tr>
<td>dsPIC30F6014A</td>
<td>60</td>
<td>22K</td>
<td>8K</td>
<td>6144</td>
<td>2048</td>
<td>5</td>
<td>16 ch</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>68</td>
<td>PT</td>
</tr>
</tbody>
</table>
Vision Sensor (Camera)

- Possible workaround on e-puck: **downsampling**
- 8 fps grayscale, 4 fps color
- Image of 1800 pixels (42x42, 80x20)
Webots: A High-Fidelity Robotic Simulator
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
In this course, we will focus on steps 2, 3 and 4 only.
Webots Features

- 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 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
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
  - Farer 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:

\[
\begin{bmatrix}
    v_L \\
    v_R
\end{bmatrix} = \begin{bmatrix}
    \alpha_{L0} & \alpha_{L1} & \cdots & \alpha_{L7} \\
    \alpha_{R0} & \alpha_{R1} & \cdots & \alpha_{R7}
\end{bmatrix} \cdot \begin{bmatrix}
    d_{IR0} \\
    \vdots \\
    d_{IR7}
\end{bmatrix} + \begin{bmatrix}
    v_{L0} \\
    v_{R0}
\end{bmatrix}
\]
Ex. 2: Artificial Neural Network

neuron N with sigmoid transfer function $f(x)$

$$O_i = f(x_i)$$

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

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

output

synaptic weight

input
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

Inhibitor: block the transmission
Suppressor: block the transmission and replace the signal with the suppressing message
Ex. 4: Behavior-Based with Subsumption

sensors \rightarrow \text{Obstacle avoidance} \rightarrow \text{Wander} \rightarrow \text{actuators}

1 \rightarrow 2 \rightarrow 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

$S_1 \rightarrow PS_1 \rightarrow MS_1$
$S_2 \rightarrow PS_2 \rightarrow MS_1$
$S_3 \rightarrow PSS_2 \rightarrow PS_3 \rightarrow MS_2$

$\Sigma \rightarrow \text{vector}$

$\rightarrow \text{motors}$

PS: Perceptual Schema
PSS: Perceptual Subschema
MS: Motor Schema
S: sensor
Ex. 5: Behavior-Based with Motor Schemas

Detect-obstacles \rightarrow\text{Avoid-obstacle} \rightarrow \text{Move-to-Goal} \rightarrow \sum \rightarrow \text{actuators}

Detect-Goal

sensors
Visualization of Vector field for Ex. 5

**Avoid-static-obstacle**

\[
V_{\text{magnitude}} = \begin{cases} 
0 & \text{for } d > S \\
\frac{S - d}{S - R} G & \text{for } R < d \leq S \\
\infty & \text{for } d \leq R
\end{cases}
\]

- \( S \) = obstacle’s sphere of influence
- \( R \) = radius of the obstacle
- \( G \) = gain
- \( d \) = distance robot to obstacle’s center

\( V_{\text{direction}} = \) radially along a line between robot and obst. center, directed away from the obstacle
Visualization of Vector field for Ex. 5

Move-to-goal (ballistic)

Output = vector = \((r, \phi)\)
(magnitude, direction)

\(V_{\text{magnitude}} = \) fixed gain value

\(V_{\text{direction}} = \) towards perceived goal
Visualization of Vector field for Ex. 5

Move-to-goal + avoid obstacle

Linear combination
(weighted sum)
Ex. 5: Behavior-Based with Motor Schemas

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 (collaborative 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


