Distributed Intelligent Systems – W3

An Introduction to Control Architectures, the e-puck Robot, and Localization Methods for Mobile Robots

Outline

- General concepts
  - Autonomy
  - Perception-to-action loop
- e-puck
  - Basic features
  - HW architecture
- Main example of reactive control arc
  - Proximal architectures
  - Distal architectures
- Localization for mobile robots
  - Positioning systems
  - Kinematic models
  - Odometry
- Localization uncertainties and navigation
  - Error sources
  - Methods for handling uncertainties
  - Odometry-based and feature-based navigation methods

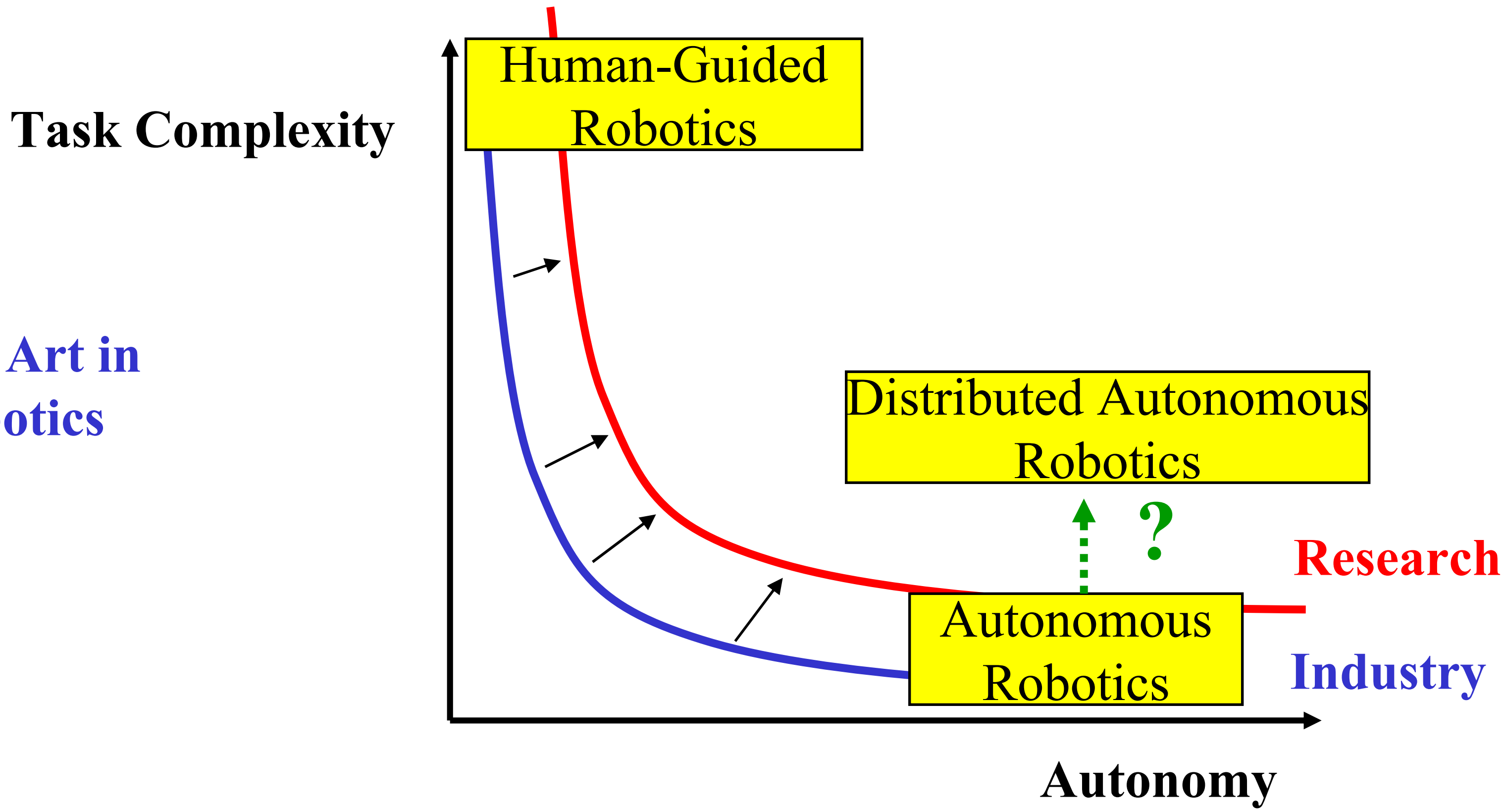
General Concepts and Principles for Mobile Robotics

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

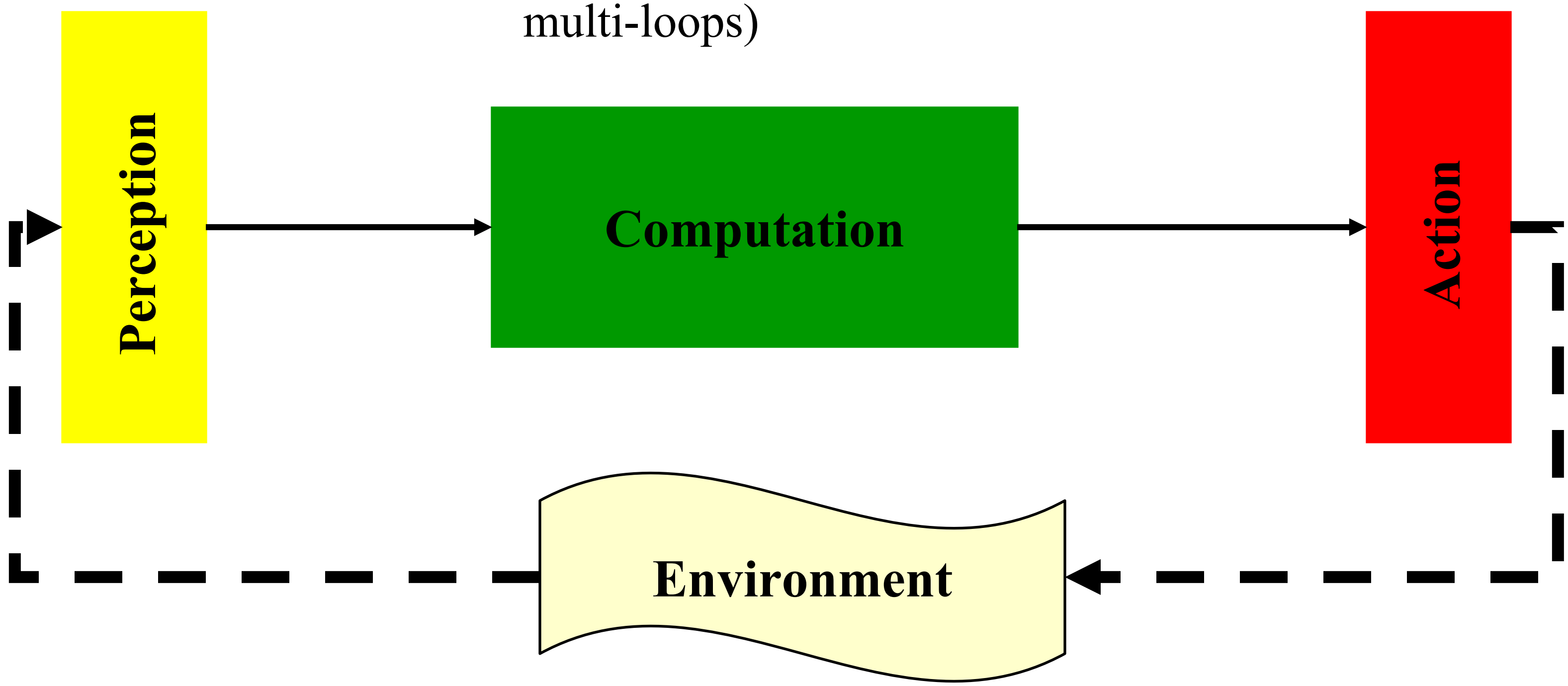


Perception-to-Action Loop

- sensors

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

- actuators



Sensors

- Propioceptive ("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 rangfinder, IR proximity sensors, ultrasound sonars

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; memory internal and potentially external to the microcontroller
- "Discretization" (analog-to-digital for values, continuous-to-discrete for time) and "continuization" (digital-to-analog for values, discrete-to-continuous for time)
- Different types of control architectures: e.g., reactive ('reflex-based") vs. deliberative ("planning")

e-puck: An Educational Robotic Tool

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

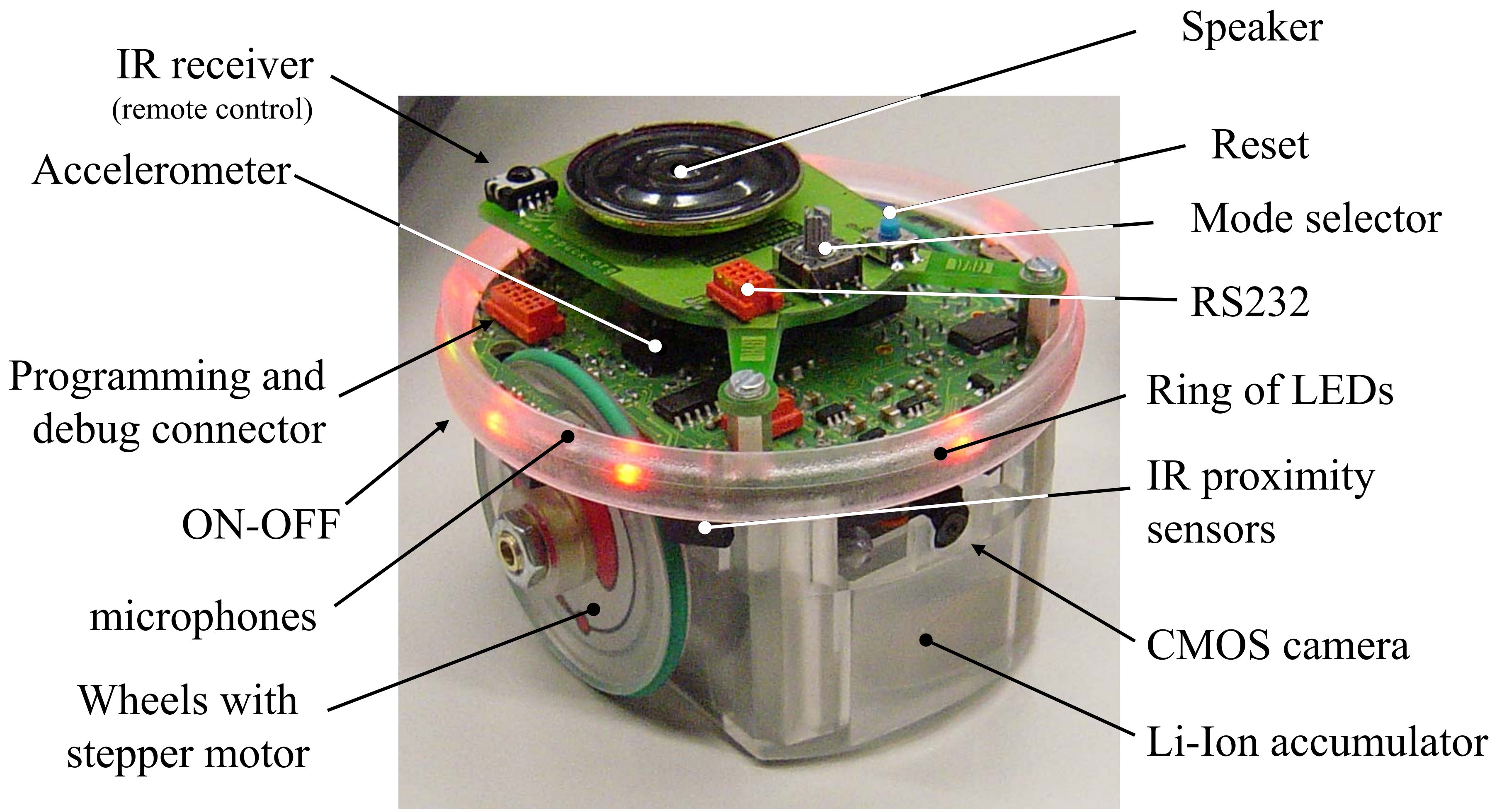
The e-puck Mobile Robot

Main features

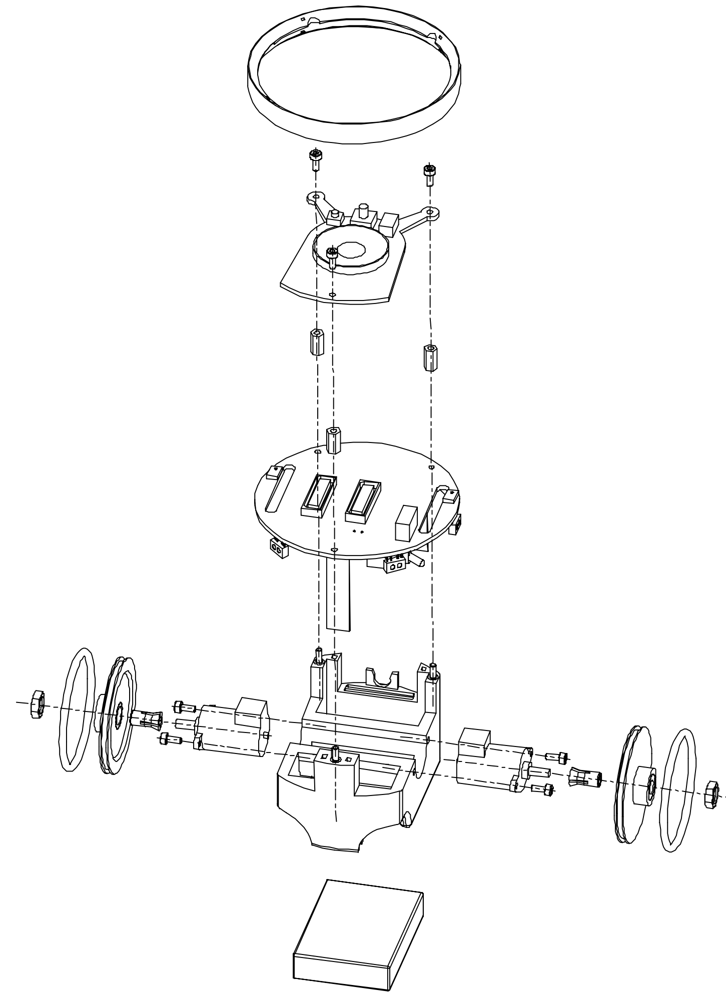
- Cylindrical, Ø 70mm
- dsPIC processor
- Two stepper motors
- Ring of LEDs
- Many sensors:
  - ✓ Camera
  - ✓ Sound
  - ✓ IR proximity
  - ✓ 3D accelerometer
- Li-ion accumulator
- Bluetooth wireless communication
- Open hardware (and software)

e-puck

e-puck Overview

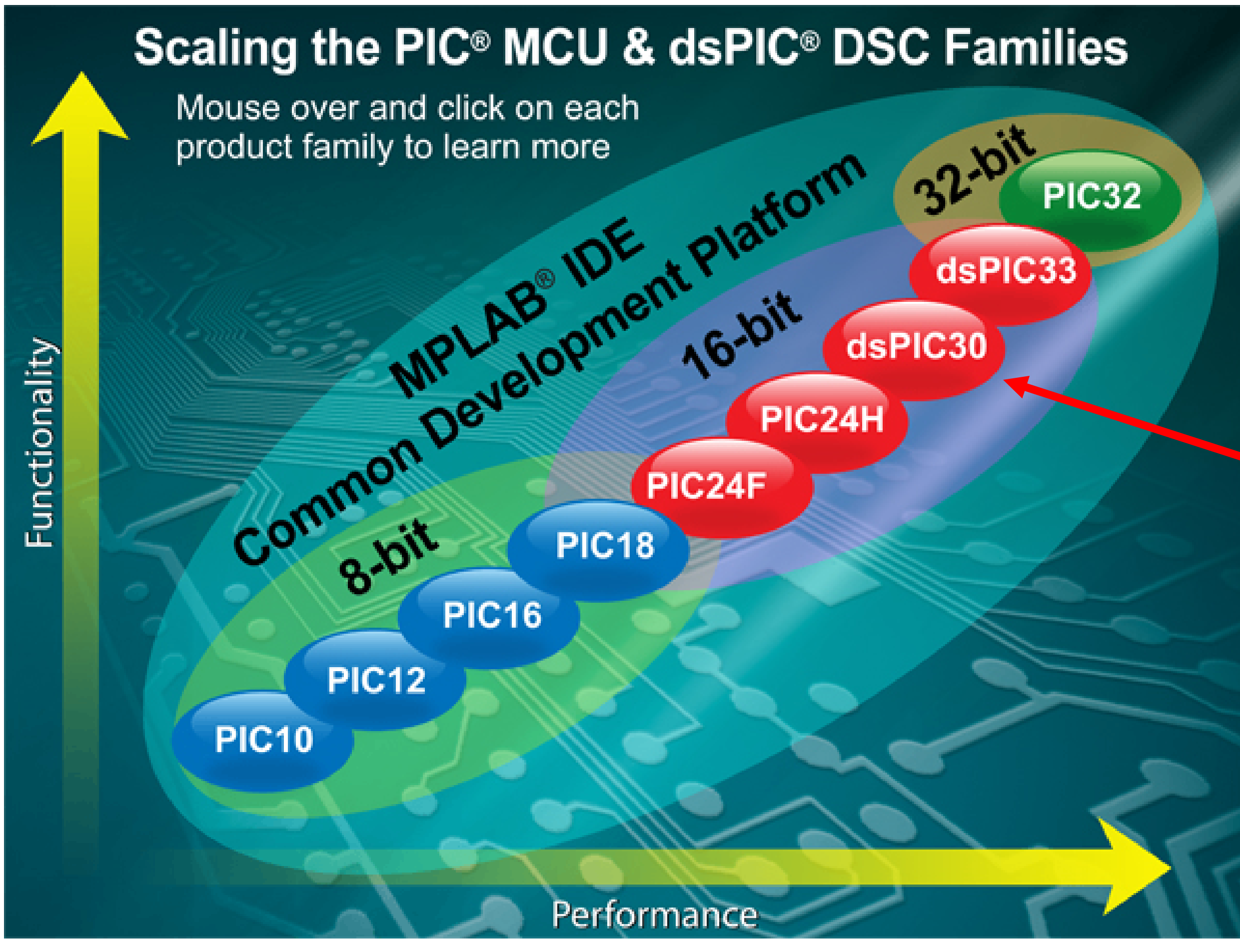


e-puck Mechanical Structure



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 Bytes	Program Memory Instructions	SRAM Bytes	EEPROM Bytes	Timer 16-bit	Input Capture	Output Compare Std. PWM	Codec Interface	A/D 12-bit 200 ksps	UART	SPI™	I²C™	CAN	I/O Pins (Max.)(1)	Packages (2)
dsPIC30F3014	40/44	24K	8K	2048	1024	3	2	2	—	13 ch	2	1	1	—	30	PG, PT
dsPIC30F4013	40/44	48K	16K	2048	1024	5	4	4	AC'97, I2S	13 ch	2	1	1	1	30	PG, PT
dsPIC30F5011	64	66K	22K	4096	1024	5	8	8	AC'97, I2S	16 ch	2	2	1	2	52	PT
dsPIC30F6011(3) dsPIC30F6011A	64	132K	44K	6144	2048	5	8	8	—	16 ch	2	2	1	2	52	PF, PT
dsPIC30F6012(3) dsPIC30F6012A	64	144K	48K	8192	4096	5	8	8	AC'97, I2S	16 ch	2	2	1	2	52	PF, PT
dsPIC30F5013	80	66K	22K	4096	1024	5	8	8	AC'97, I2S	16 ch	2	2	1	2	68	PT
dsPIC30F6013(3) dsPIC30F6013A	80	132K	44K	6144	2048	5	8	8	—	16 ch	2	2	1	2	68	PF, PT
dsPIC30F6014(3) dsPIC30F6014A	80	144K	48K	8192	4096	5	8	8	AC'97, I2S	16 ch	2	2	1	2	68	PF, PT

e-puck microcontroller

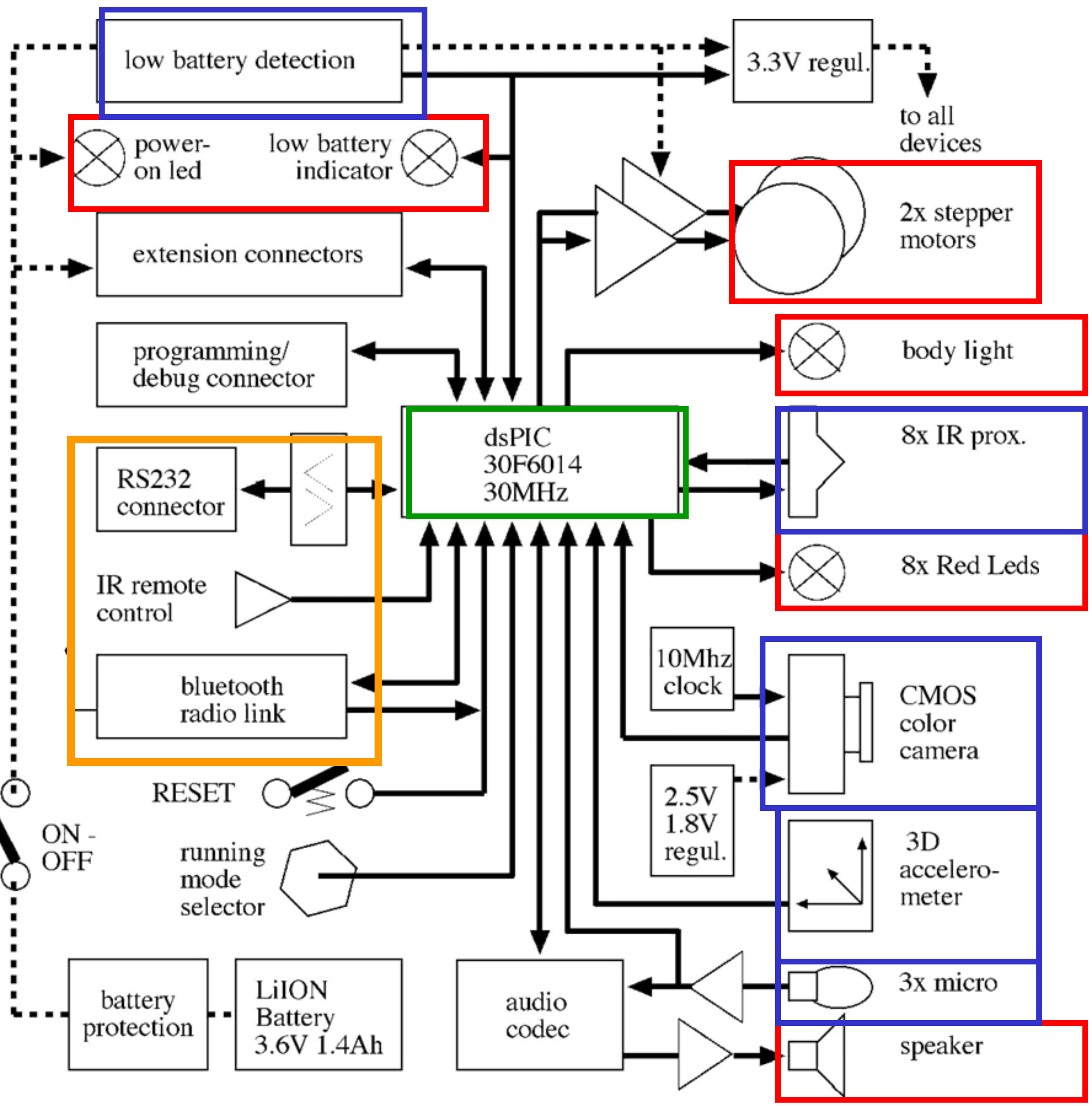
e-Puck Block Schema

Actuators?

Sensors?

Computation?

Communication?



e-puck Accelerometer

- Sampling of the continous time analog accelerometer (3 axes) using the integrated A/D converter
- Low to medium sampling frequency; typically a function of the application and of the accelerometer characteristics

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

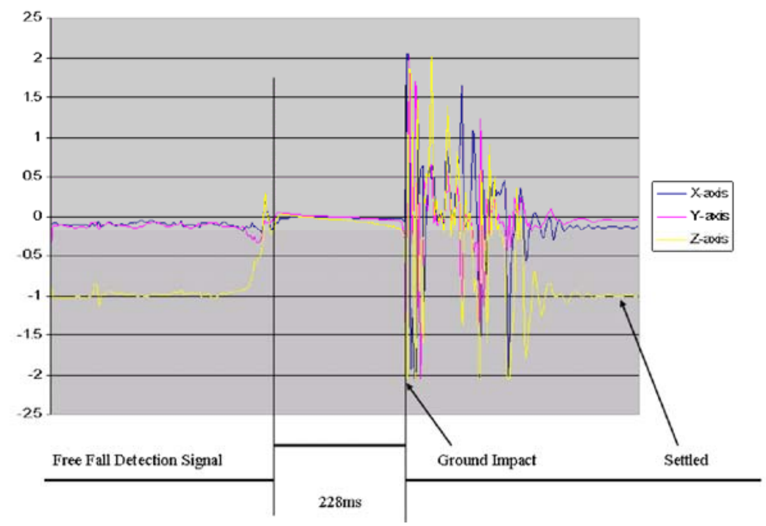


Table 2. Operating Characteristics

Unless otherwise noted: $-20°C \le T_A \le 85°C$, $2.2\ V \le V_{DD} \le 3.6\ V$, Acceleration = 0g, Loaded output[1]

Characteristic	Symbol	Min	Typ	Max	Unit
Operating Range[2]					
Supply Voltage[3]	V_{DD}	2.2	3.3	3.6	V
Supply Current	I_{DD}	—	500	800	µA
Supply Current at Sleep Mode[4]	I_{DD}	—	3	10	µA
Operating Temperature Range	T_A	–20	—	+85	°C
Acceleration Range, X-Axis, Y-Axis, Z-Axis					
g-Select1 & 2: 00	g_{FS}	—	±1.5	—	g
g-Select1 & 2: 10	g_{FS}	—	±2.0	—	g
g-Select1 & 2: 01	g_{FS}	—	±4.0	—	g
g-Select1 & 2: 11	g_{FS}	—	±6.0	—	g
Output Signal					
Zero g (T_A = 25°C, V_{DD} = 3.3 V)[5]	V_{OFF}	1.485	1.65	1.815	V
Zero g	V_{OFF}, T_A	—	±2	—	mg/°C
Sensitivity (T_A = 25°C, V_{DD} = 3.3 V)					
1.5g	$S_{1.5g}$	740	800	860	mV/g
2g	S_{2g}	555	600	645	mV/g
4g	S_{4g}	277.5	300	322.5	mV/g
6g	S_{6g}	185	200	215	mV/g
Sensitivity	S, T_A	—	±3	—	%/°C
Bandwidth Response					
XY	f_{-3dB}	—	350	—	Hz
Z	f_{-3dB}	—	150	—	Hz

Freescale Semiconductor
Technical Data

MMA7260Q
Rev 0, 04/2005

MMA7260Q

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

The MMA7260Q low cost capacitive micromachined accelerometer features signal conditioning, a 1-pole low pass filter, temperature compensation and g-Select which allows for the selection among 4 sensitivities. Zero-g offset full scale span and filter cut-off are factory set and require no external devices. Includes a Sleep Mode that makes it ideal for handheld battery powered electronics.

Features

- Selectable Sensitivity (1.5g/2g/4g/6g)
- Low Current Consumption: 500 µA
- Sleep Mode: 3 µA
- Low Voltage Operation: 2.2 V – 3.6 V
- 6mm x 6mm x 1.45mm QFN
- High Sensitivity (800 mV/g @1.5 g)
- Fast Turn On Time
- High Sensitivity (1.5 g)
- Integral Signal Conditioning with Low Pass Filter
- Robust Design, High Shocks Survivability
- Pb-Free Terminations
- Environmentally Preferred Package
- Low Cost

MMA7260Q: XYZ AXIS ACCELEROMETER ±1.5g/2g/4g/6g

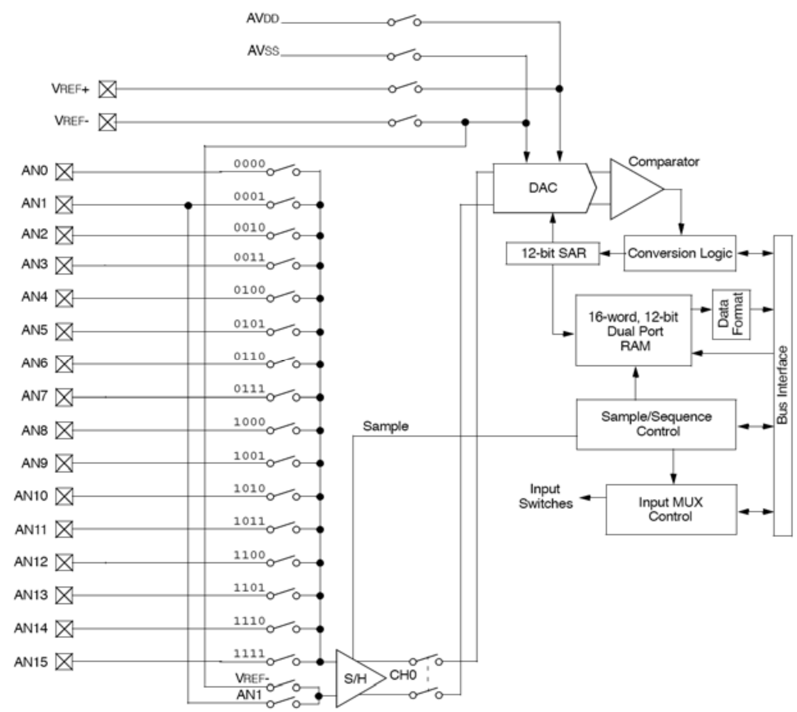
Bottom View

16 LEAD
QFN
CASE 1622-01

e-puck Hearing Capabilities

Example: acoustic source localization

- medium to high sampling frequency application
- E.g.: robot dimension 7.5 cm → microphone max inter-distance → 5.5 cm → speed of sound in air 340 m/s → travel time micro-to-micro 0 (orthogonal) to 160 µs (aligned) → 6 kHz min to max possible on the device
- max DsPIC sampling frequency (1 channel): 200 KHz (see datasheet)
- 2 micros: 2 ch. e.g. 85 kHz → 12 µs → 4 mm resolution but possible aliasing on a plane (dual localization)
- 3 micros: 3 ch., e.g. 56 kHz → 18 µs → 6 mm but no aliasing on a plane (unique localization)



e-puck Vision Capabilities

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 x 480 x 3 x 30 = 27Mbytes/second
- The dsPIC can execute max 15MIPS (millions of instructions/second)

Memory

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

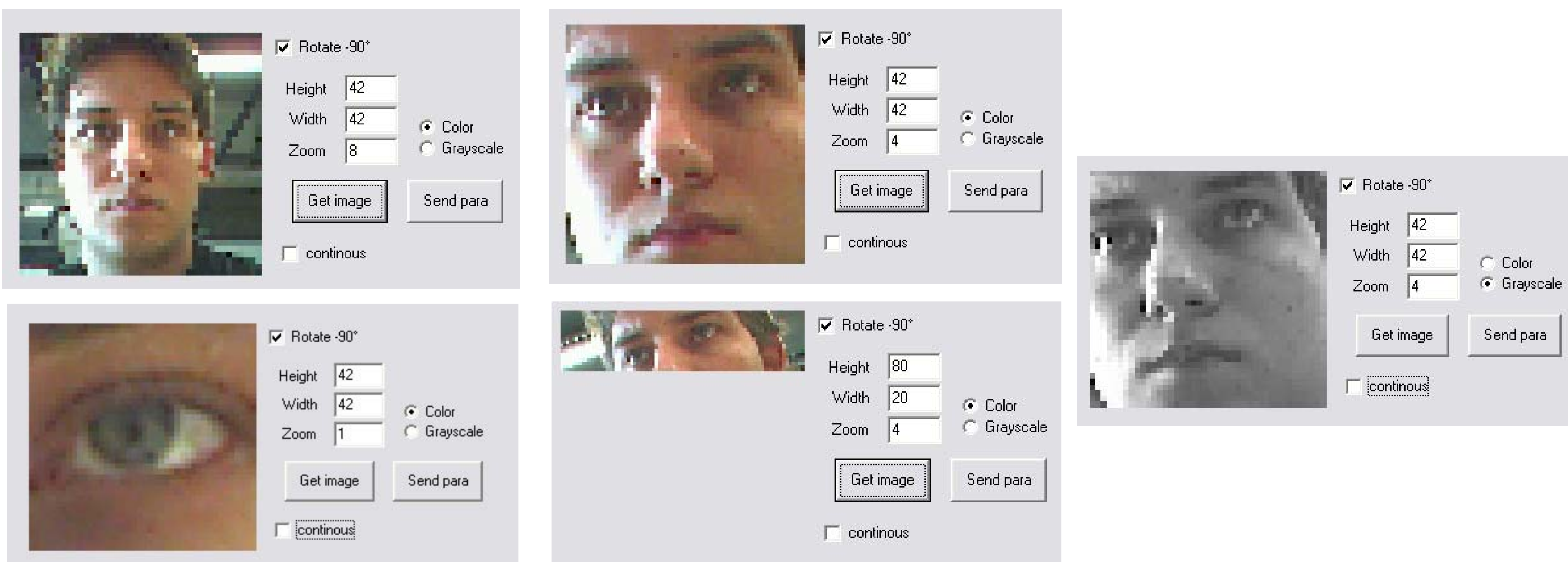
TABLE 1-1: dsPIC30F GENERAL PURPOSE FAMILY VARIANTS

Device	Pins	Program Memory		SRAM Bytes	EEPROM Bytes	Timer 16-bit	Input Capture	Output Compare Std. PWM	Codec Interface	A/D 12-bit 200 ksps	UART	SPI™	I²C™	CAN	I/O Pins (Max.)[1]	Packages[2]
		Bytes	Instructions													
dsPIC30F3014	40/44	24K	8K	2048	1024	3	2	2	—	13 ch	2	1	1	—	30	PG, PT
dsPIC30F4013	40/44	48K	16K	2048	1024	5	4	4	AC'97, I2S	13 ch	2	1	1	1	30	PG, PT
dsPIC30F5011	64	66K	22K	4096	1024	5	8	8	AC'97, I2S	16 ch	2	2	1	2	52	PT
dsPIC30F6011[3] dsPIC30F6011A	64	132K	44K	6144	2048	5	8	8	—	16 ch	2	2	1	2	52	PF, PT
dsPIC30F6012[3] dsPIC30F6012A	64	144K	48K	8192	4096	5	8	8	AC'97, I2S	16 ch	2	2	1	2	52	PF, PT
dsPIC30F5013	80	66K	22K	4096	1024	5	8	8	AC'97, I2S	16 ch	2	2	1	2	68	PT
dsPIC30F6013[3] dsPIC30F6013A	80	132K	44K	6144	2048	5	8	8	—	16 ch	2	2	1	2	68	PF, PT
dsPIC30F6014[3] dsPIC30F6014A	80	144K	48K	8192	4096	5	8	8	AC'97, I2S	16 ch	2	2	1	2	68	PF, PT

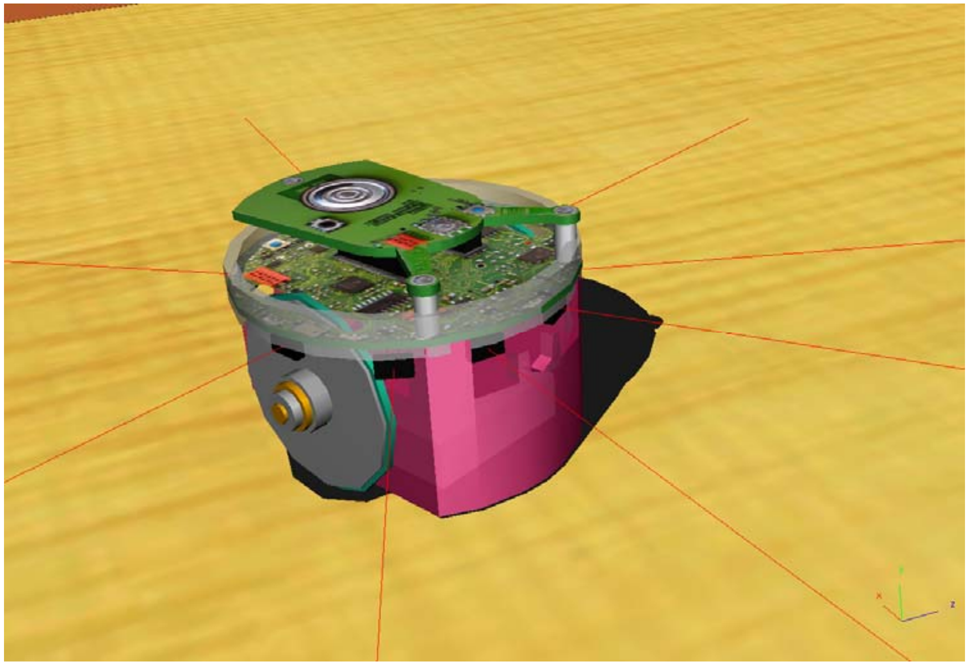
e-puck microcontroller →

e-puck Vision Capabilities

- Possible workaround on e-puck: downsampling
- 8 fps grayscale, 4 fps color
- Image of 1800 pixels (42x42, 80x20)



Real and Simulated e-Puck



Real e-Puck

Realistically simulated e-Puck (Webots)

- sensor- and actuator-based
- noise, nonlinearities of S&A reproduced
- kinematic (e.g., speed, position) and dynamic (e.g., mass, forces, friction,)

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

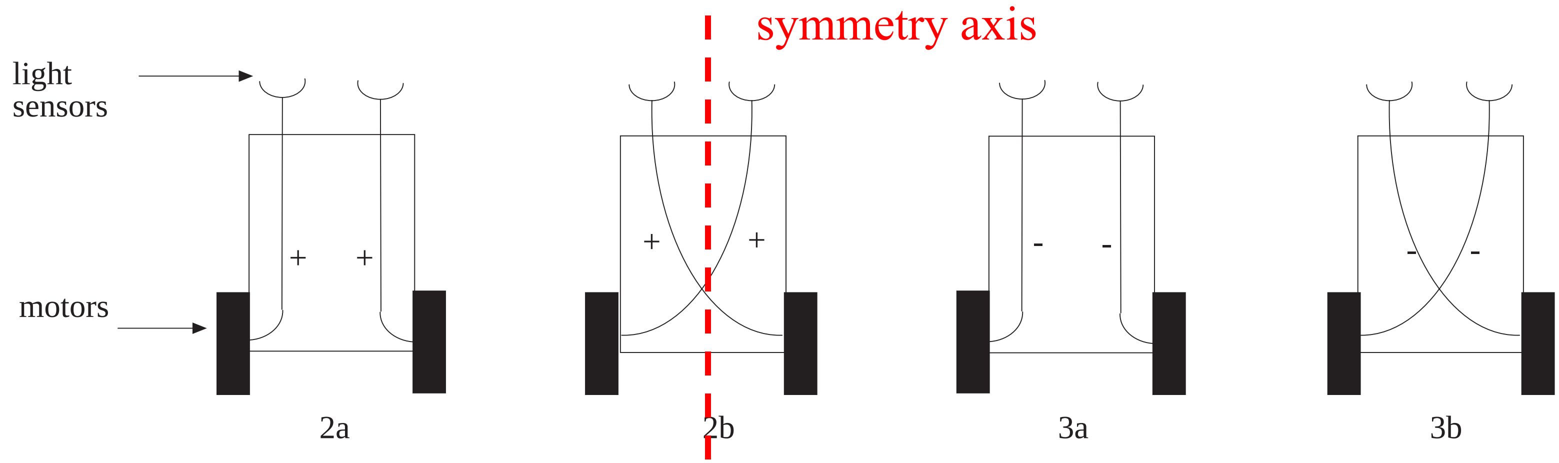
Reactive Architectures: Proximal vs. Distal in Theory

- Distal architectures
  - Farer from sensor and actuators
  - More elaborated data processing (e.g., filtering)
  - 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

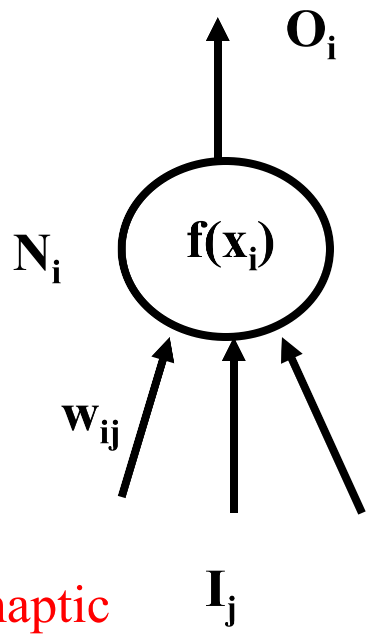
- A whole blend!
- Four "classical" examples of reactive control architecture for solving the same problem: obstacle avoidance.
- Two proximal: Braitenberg and Artificial Neural Network
- Two distal: Subsumption and Motor Schema, both behavior-based

Ex. 1: Braitenberg's Vehicles



- Work on the difference (gradient) between sensors
- Originally omni-directional sensors but work even better with directional sensors
- + excitation, - inibition; linear controller (out = signed coefficient * in)
- Symmetry axis along main axis of the vehicle (----)
- Originally: light sensors; works perfectly also with proximity sensors (3c?)

Ex. 2: Artificial Neural Network

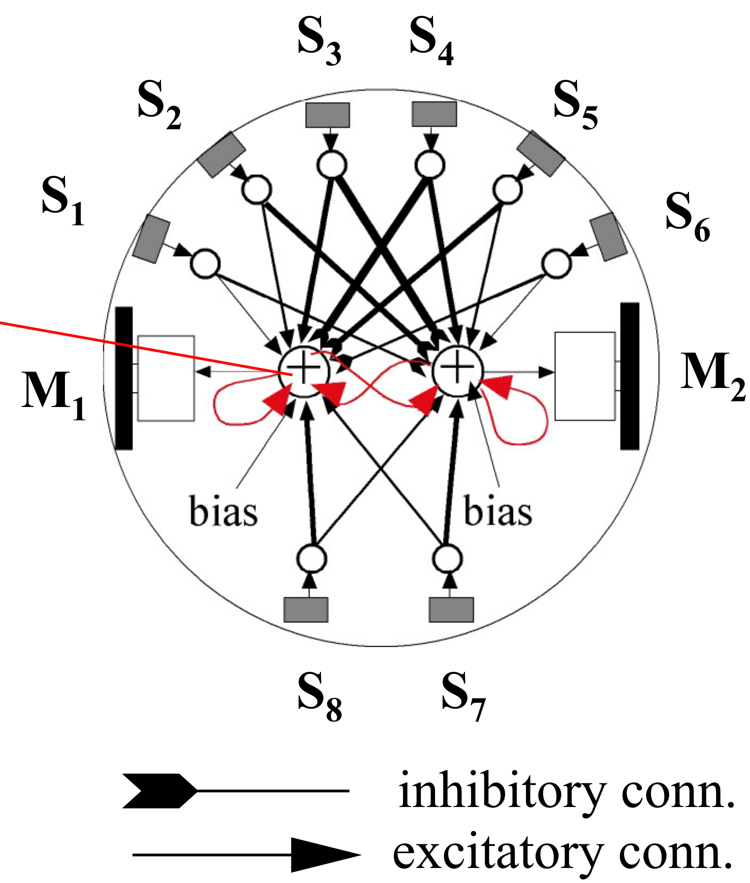


neuron N with sigmoid transfer function f(x)

$$O_i = f(x_i)$$

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

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



Ex. 3: Rule-Based

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

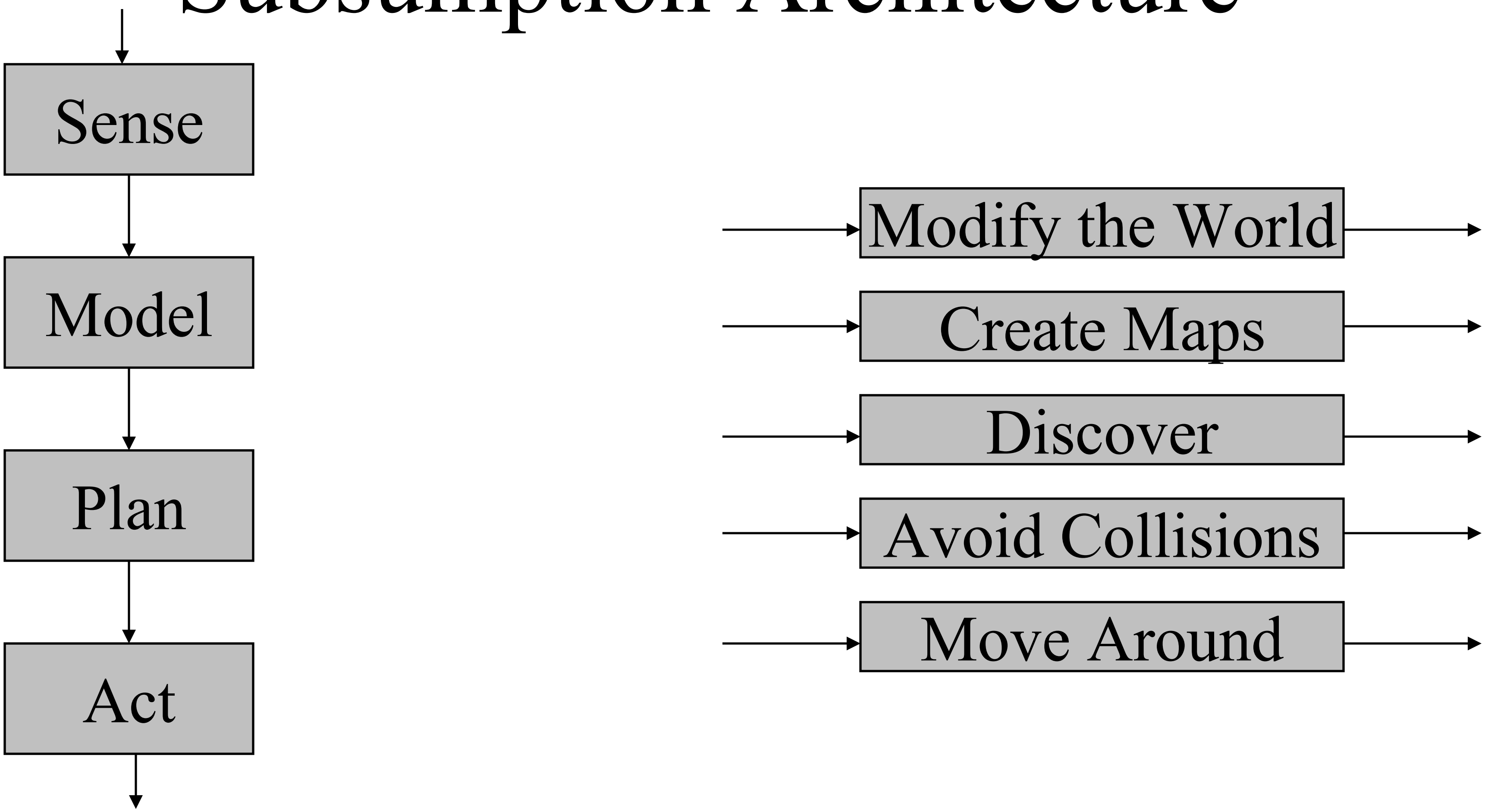
Rule 2:
if (proximity sensors on the right active) **then**
turn left

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)

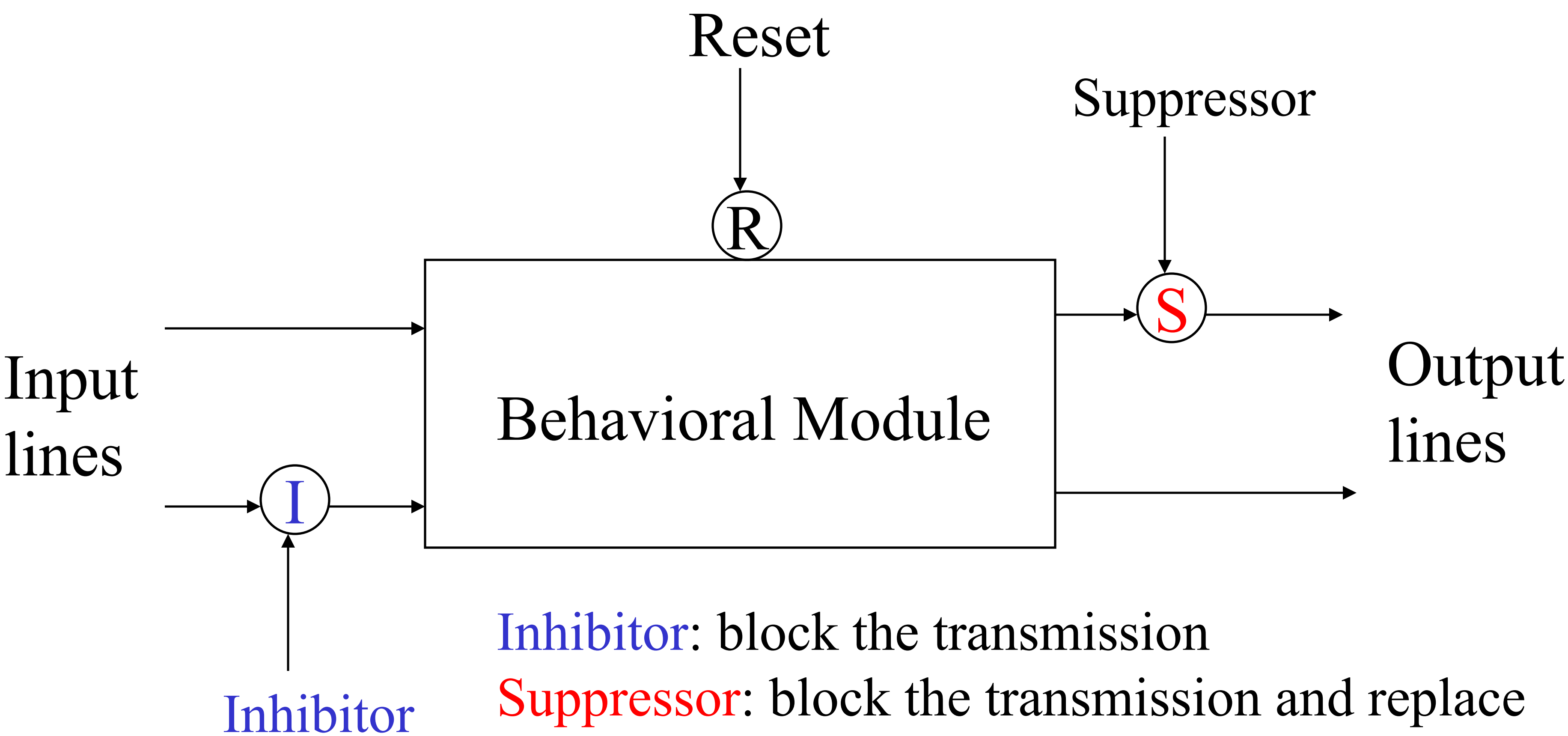
Subsumption Architecture



Classical paradigm (serial); emphasis on deliberative control

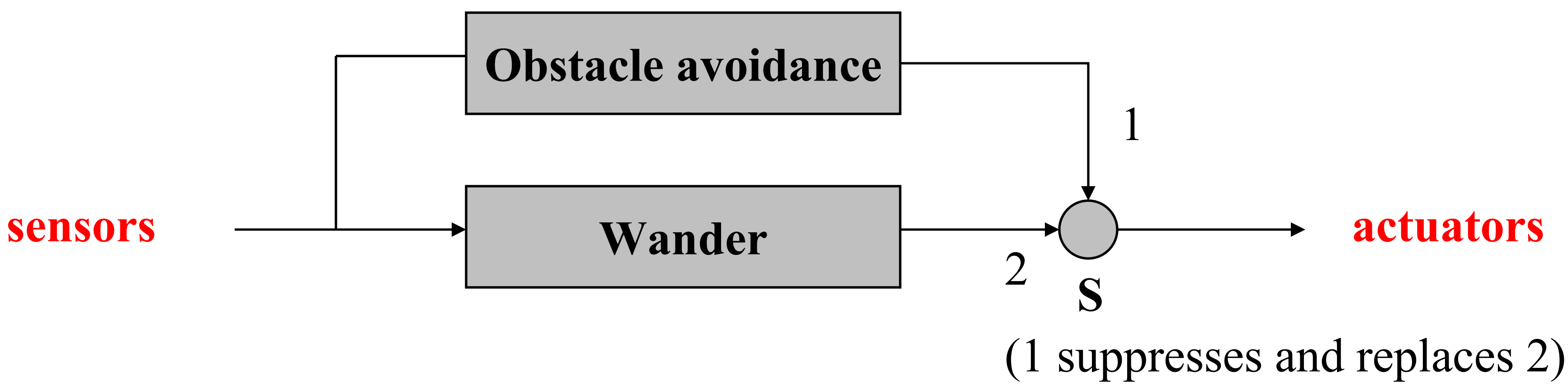
Subsumption (parallel); emphasis on reactive control

Subsumption Architecture: AFSM



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

Ex. 4: Behavior-Based with Subsumption



Evaluation of Subsumption

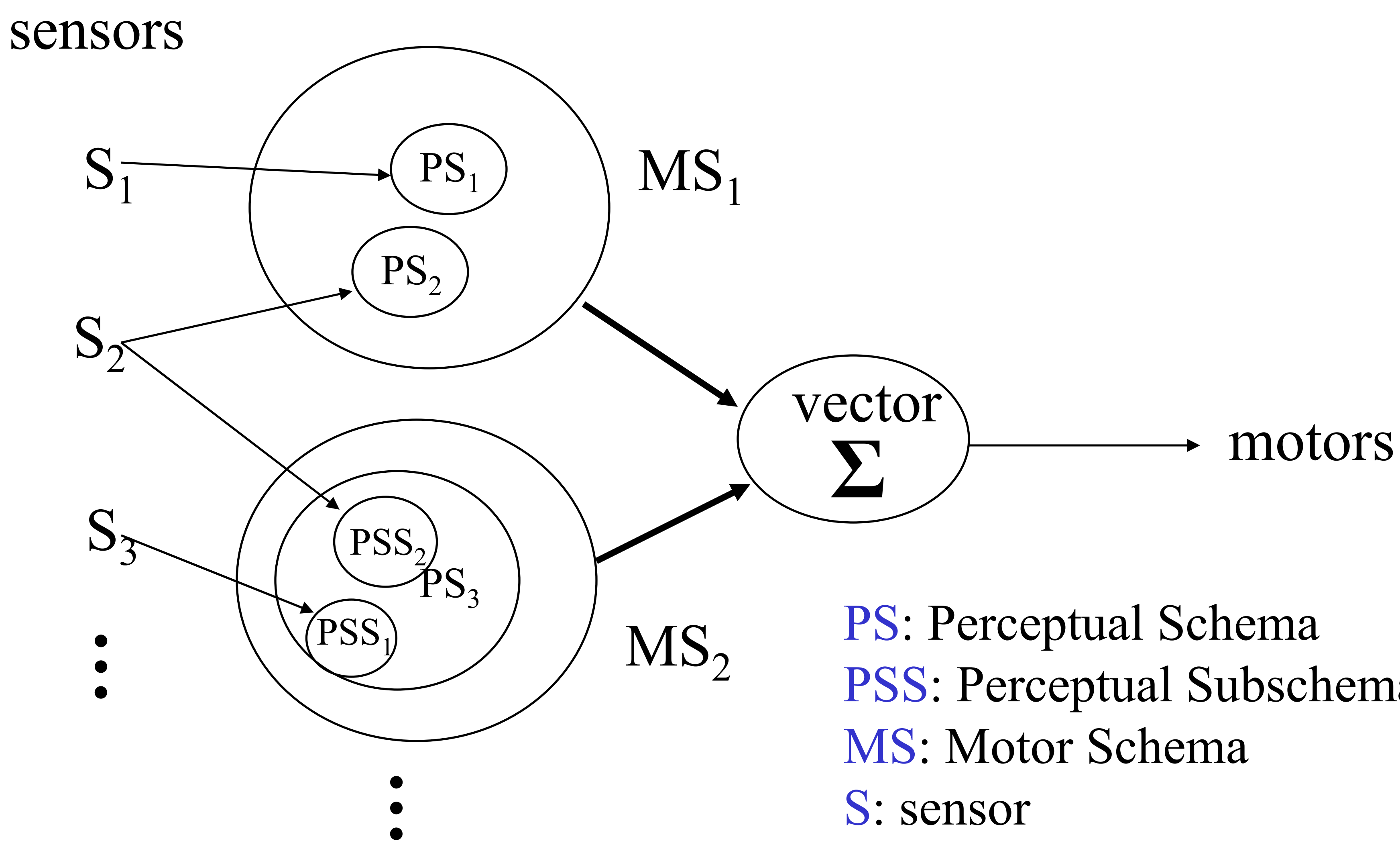
+ Support for parallelism: each behavioral layer can run independently and asynchronously (including different loop time)
+ HW retargetability: can compile down directly to programmable-array logic circuitry

- Hardwiring mean less run time flexibility
- 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



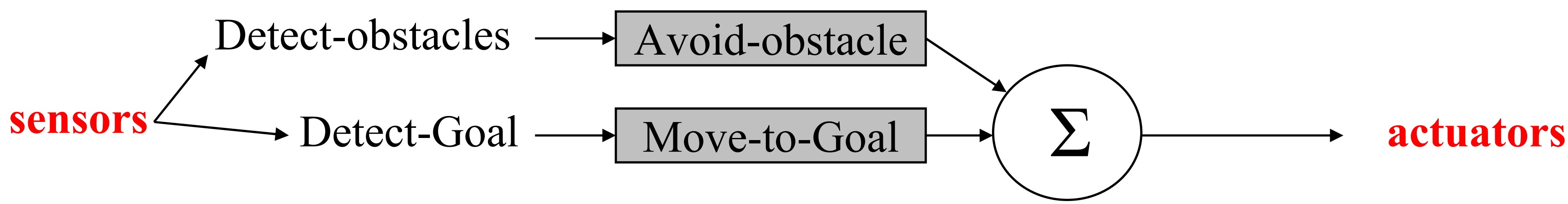
PS: Perceptual Schema

PSS: Perceptual Subschema

MS: Motor Schema

S: sensor

Ex. 5: Behavior-Based with Motor Schemas



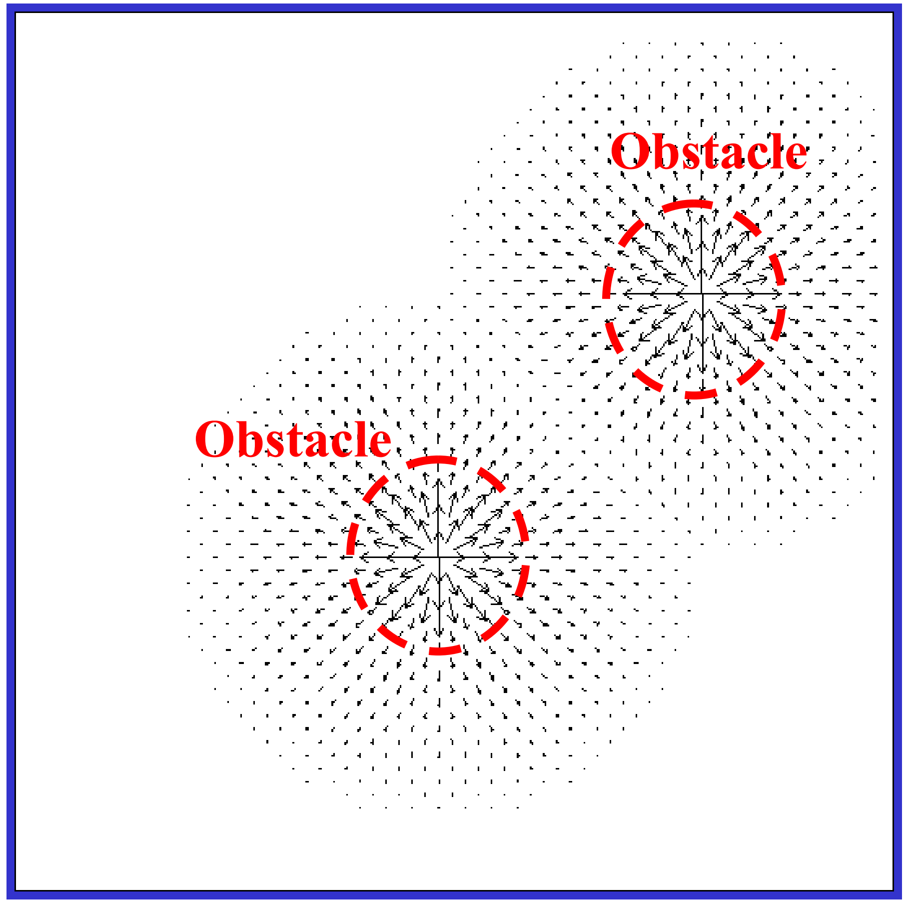
Visualization of Vector field for Ex. 5

Avoid-static-obstacle

$$V_{\text{magnitude}} = \begin{cases} 0 & \textit{for} \quad d > S \\ \dfrac{S-d}{S-R} G & \textit{for} \quad R < d \leq S \\ \infty & \textit{for} \quad d \leq R \end{cases}$$

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



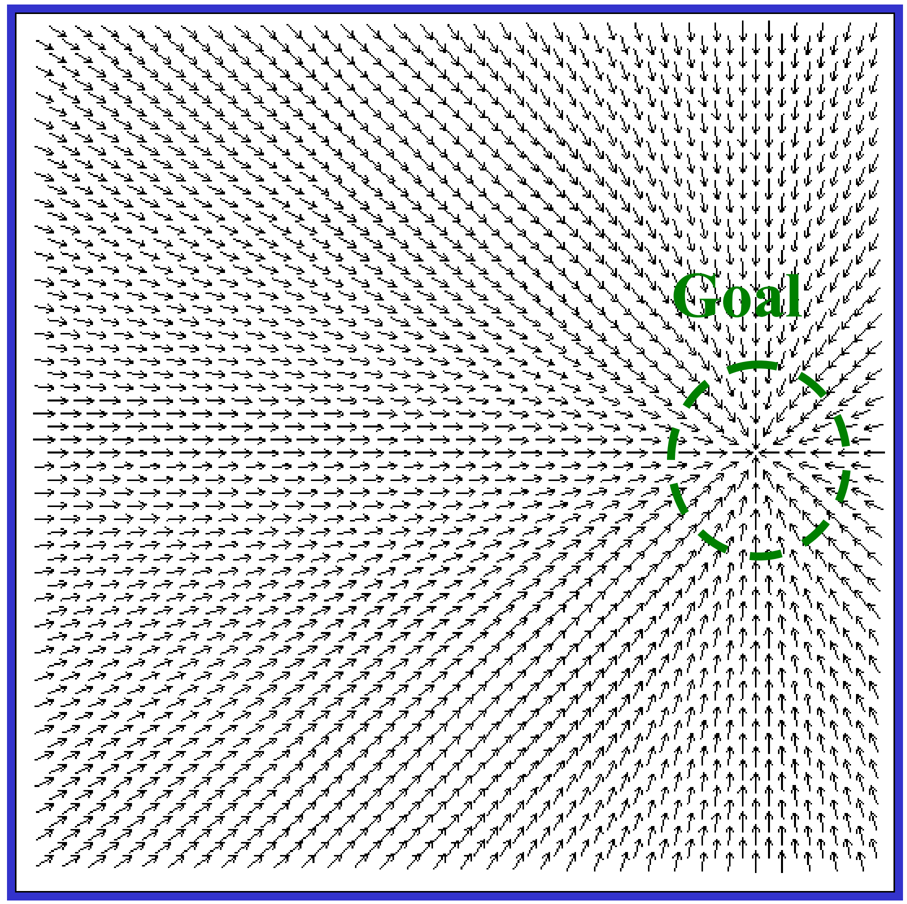
Visualization of Vector field for Ex. 5

Move-to-goal (ballistic)

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

$V_{magnitude}$ = fixed gain value

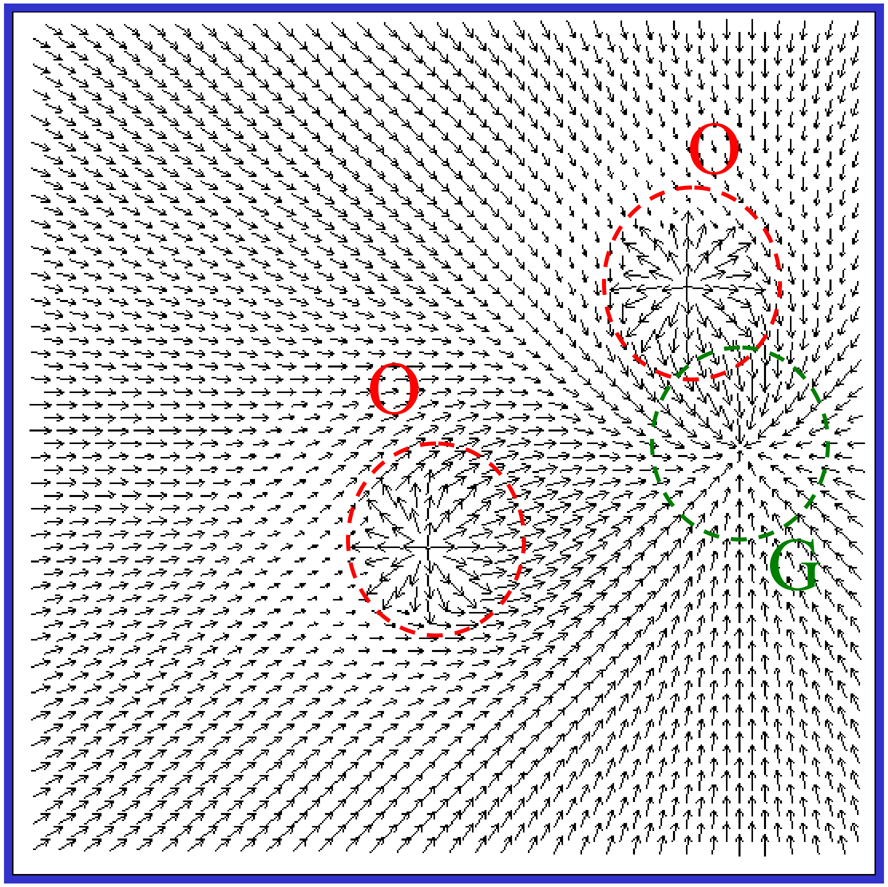
$V_{direction}$ = towards perceived goal



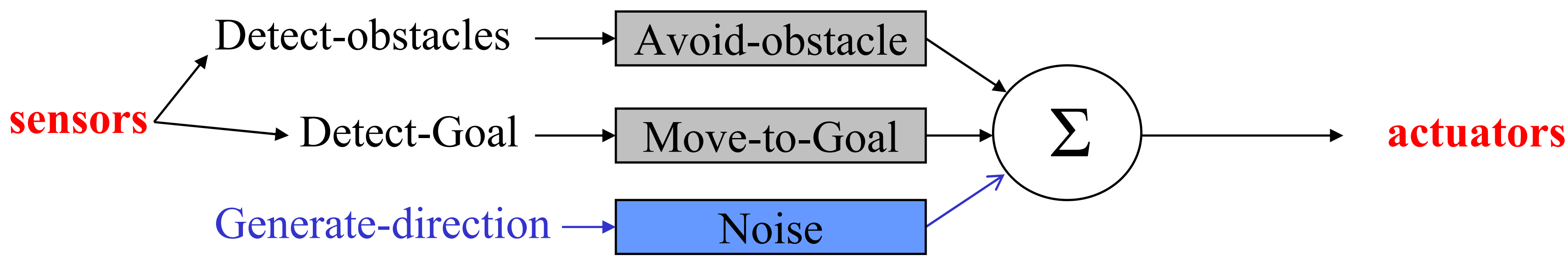
Visualization of Vector field for Ex. 5

Move-to-goal + avoid obstacle

Linear combination
(weigthed sum)



Ex. 5: Behavior-Based with Motor Schemas



For avoiding to get stuck in local minima
(typical problem of vector field approaches)

Evaluation of Motor Schemas

+ Support for parallelism: motor schemas are naturally parallelizable
+ Run time flexibility: schemas = software agents -> reconfigurable on the flight

- Robustness -> well-known problems of potential field approach -> extra introduction of noise (not clear method for exploiting that generated by sensors, …)
- Slow and computationally expensive sometimes
- No HW retargetability: do not provide HW compilers; do not take into account the system as a whole

Evaluation of both Architectures in Practice

- In pratice (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, …
- Pierre Arnaud's work (thesis and book EPFL, 2000, see references at the end); Masoud Asadpour's work (thesis EPFL, 2006, see reference at the end) went in this direction for different reasons

Robot Localization and Positioning Systems

Classification axes

- Indoor vs. outdoor techniques
- Absolute vs. relative positioning systems
- Line-of-sight vs. obstacle passing/surrounding
- Underlying physical principle and channel
- Positioning available on-board vs. off-board
- Scalability in terms of number of nodes

Performance of Positioning Systems

- As any another sensor, “position sensor”
- accuracy, precision, range, positioning update frequency

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

m = measured value
v = true value

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

σ = standard dev of the sensor noise

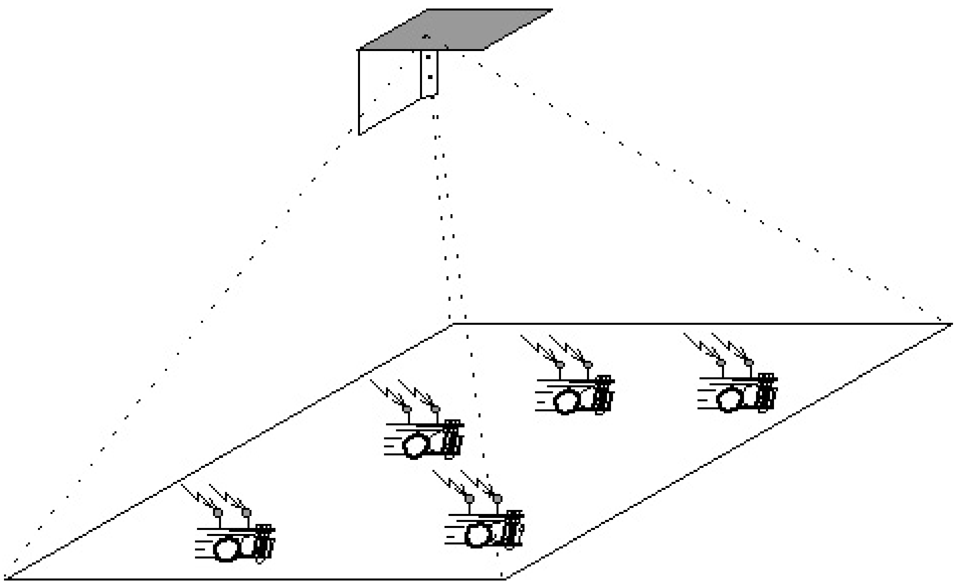
[From *Introduction to Autonomous Mobile Robots*, Siegwart R. and Nourbakhsh I. R.]

Indoor Positioning Systems

Selected Indoor Positioning Systems

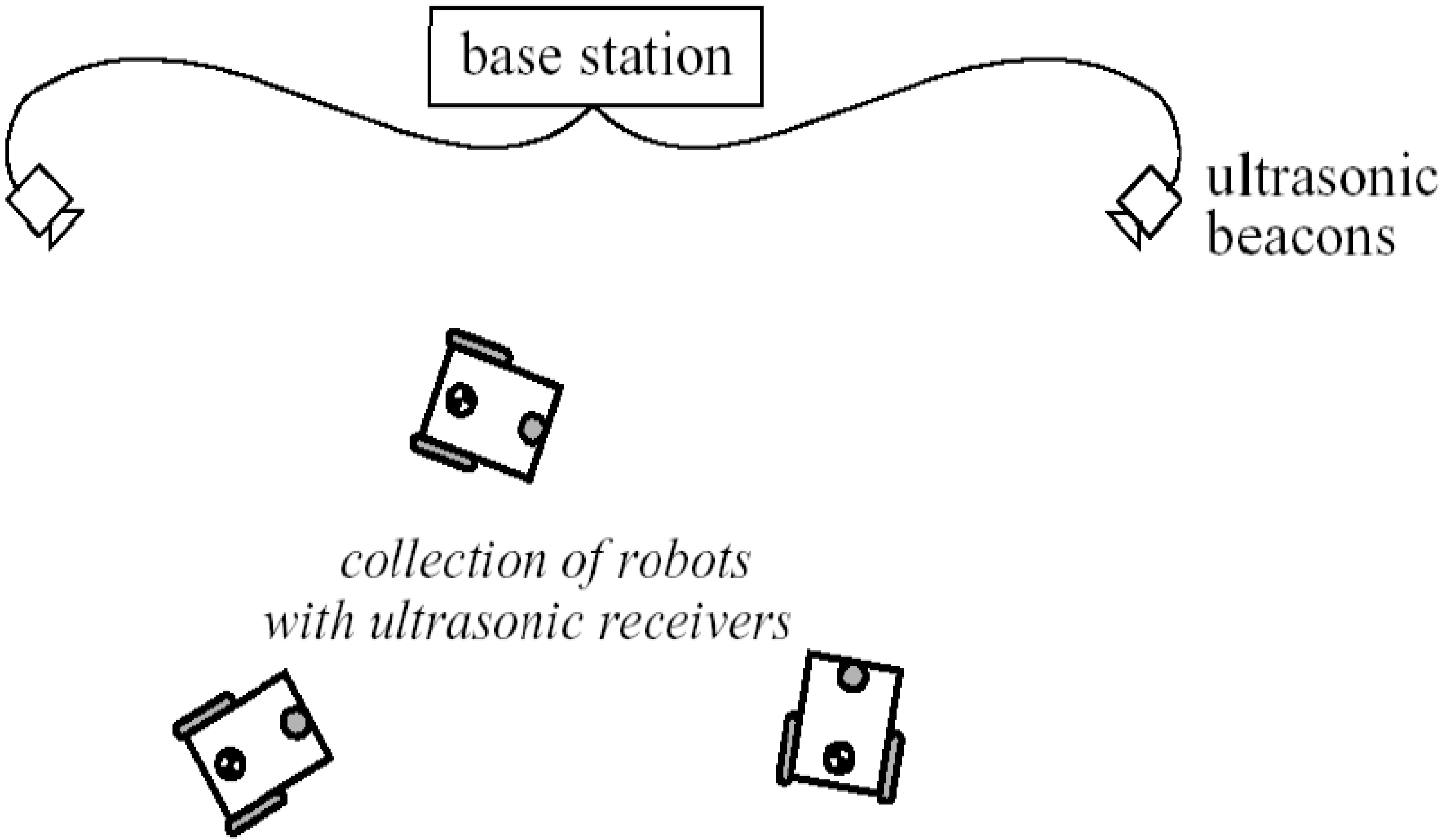
- Laser-based indoor GPS
- Ultrasound (US) + radio frequency (RF) technology
- Infrared (IR) + RF technology
- Vision-based overhead system
- Impulse Radio Ultra Wide Band (IR-UWB)

Laser-Based Indoor (KPS)



- Performance: a few mm in position over 5x5 m arena, 25-50 Hz, a few degrees in orientation
- Position available on the robot without com (GPS-like)
- Line-of-Sight (LOS) method
- Tested in 2D but extensible in 3D (2 laser base stations)

Ultrasound + Radio Technology



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

Ultrasound + Radio Technology

- Principle: time of arrival on 3 (2D) or 4 (3D) US receptors, synchronization with radio signal
- Used for relative (on the robots) and absolute positioning (fixed beacons)
- Accuracy: sub cm accuracy over several m for a 30 cm radius platform (e.g. Michaud et al, ICRA 2008)
- Accuracy inversely proportional with size of the module (proportional to distance between US receptors)
- Updating speed: 1/(0.075*N_robots) Hz (e.g., < 1 Hz with 14 or more robots) (Michaud et al, ICRA 2008)
- Better than LOS but obstacle influence sound propagation

Infrared + Radio Technology

- Principle:
  - belt of IR emitters (LED) and receivers (photodiode)
  - IR LED used as antennas; modulated light (carrier 10.7 MHz), RF chip behind
  - Range: measurement of the Received Signal Strength Intensity (RSSI)
  - Bearing: signal correlation over multiple receivers
  - Measure range & bearing can be coupled with standard RF channel (e.g. 802.11) for heading assessment
  - Can also be used for 20 kbit/s IR com channel
  - Robot ID communicated with the IR channel (ad hoc protocol)

[Pugh et al., *IEEE Trans. on Mechatronics*, 2009]

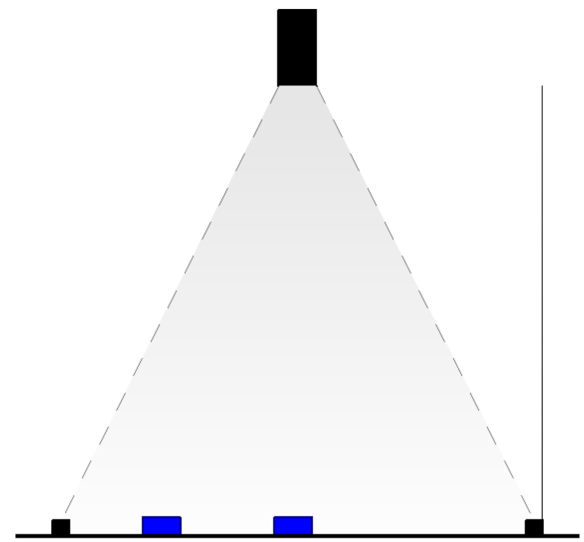
Infrared + Radio Technology

Performance summary:

- Range: 3.5 m
- Update frequency 25 Hz with 10 neighboring robots (or 250 Hz with 2)
- Accuracy range: <7% (MAX), generally decrease 1/d
- Accuracy bearing: < 9º (RMS)
- LOS method
- Possible extension in 3D, larger range (but more power) and better bearing accuracy with more photodiodes (e.g. Bergbreiter, PhD UCB 2008, dedicated asic, up to 15 m, 256 photodiodes, single emitter with conic lense)

Overhead (Multi-)Camera Systems

- Tracking objects with one (or more) overhead cameras
- Absolute positions, available outside the robot/sensor
- Active, passive, or no markers
- Open source software
- Major issues: light, calibration
- E.g. open-source software SwisTrack (developed at DISAL)



Accuracy	~ 1 cm (2D)
Update rate	~ 20 Hz
# agents	~ 100
Area	~ 10 m^2

IR-UWB System (e.g. Ubisense)

- Tracking UWB tags
- Absolute positions, available outside the robot/sensor
- Multiple antennas
- Battery for 5 years
- 6 - 8 GHz UWB channel
- Issue: because of multi-path and multi-user interference performances (accuracy and update rate) significantly degraded

Accuracy	15 cm (3D)
Update rate	40 Hz / tag
# agents	~ 10000
Area	~ 1000 m^2

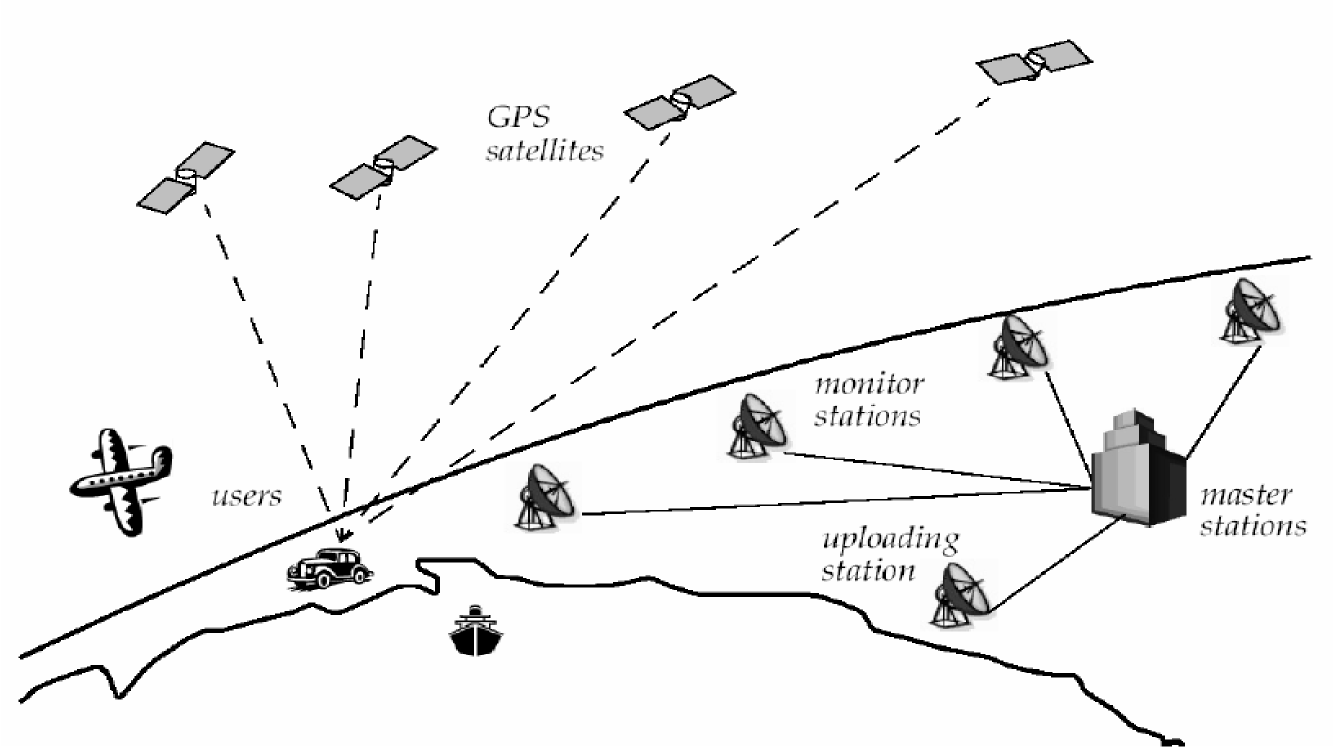
Outdoor Positioning Systems

Selected Outdoor Positioning Techniques

- GPS
- Differential GPS (dGPS)

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Global Positioning System



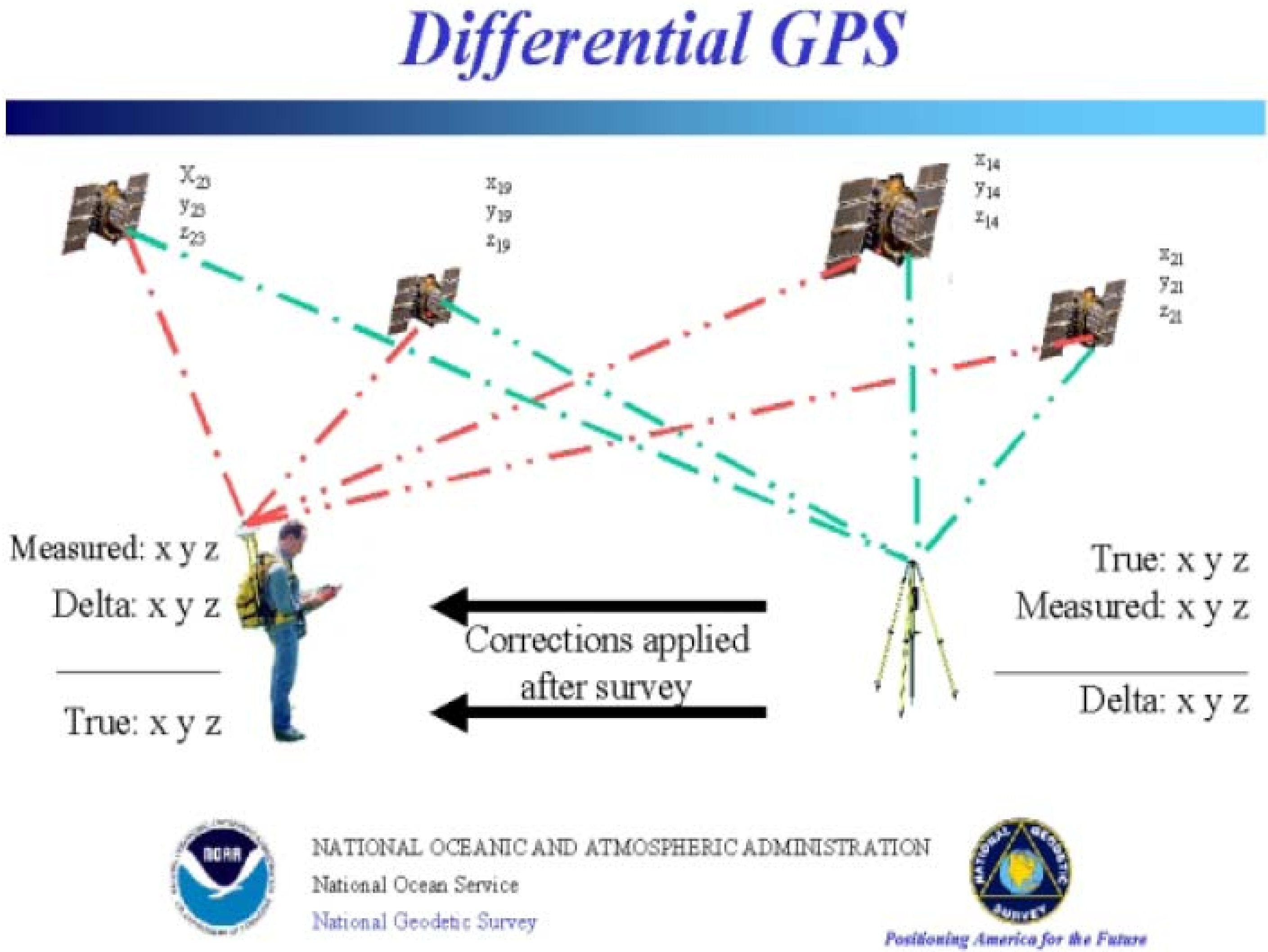
© R. Siegwart, ETH Zurich - ASL

Global Positioning System

- 24 satellites (including three spares) orbiting the earth every 12 hours at a height of 20.190 km.
- Satellites synchronize their transmission so that signals are broadcasted at the same time (ground stations updating + atomic clocks)
- Location of any GPS receiver is determined through a time of flight measurement
- Real time update of the exact location of the satellites:
  - monitoring the satellites from a number of widely distributed ground stations
  - master station analyses all the measurements and transmits the actual position to each of the satellites
- Exact measurement of the time of flight
  - the receiver correlates a pseudocode with the same code coming from the satellite
  - The delay time for best correlation represents the time of flight.
  - quartz clock on the GPS receivers are not very precise
  - the range measurement with (at least) four satellites allows to identify the three values (x, y, z) for the position and the clock correction ΔT
- Recent commercial GPS receiver devices allows position accuracies down to a couple meters.

dGPS

Position accuracy: typically from a few to a few tens of cm



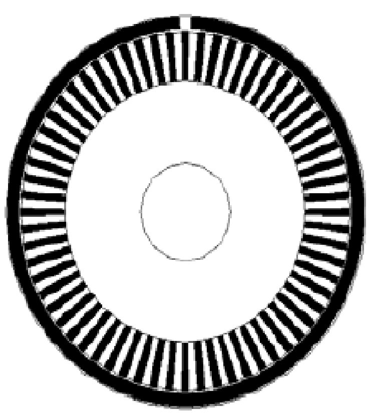
Odometry for Differential-Wheel Vehicles

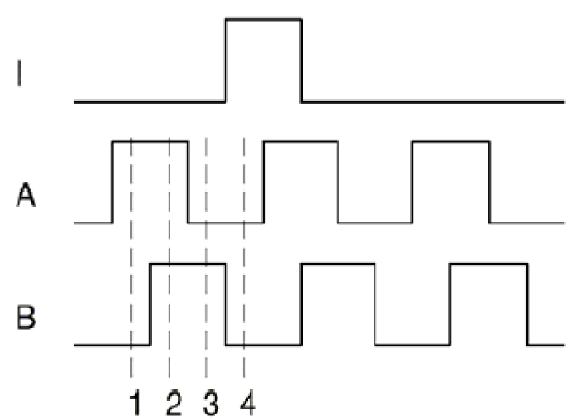
Odometry: Idea and Motivation

- Positioning (and orientation) for a mobile robot is key
- Q: can we track the absolute position and orientation (global/environmental reference frame) based on movement information exclusively measured by on-board proprioceptive information?
- A: yes, using odometry! (and knowledge of initial position and orientation)
- Needed: propioceptive movement sensors such as
  - DC motors + encoders (closed-loop control)
  - motor step counters (open-loop control of stepper motors but pre-established fixed increment per pulse, as on e-puck)
  - accelerometers (e-puck has a 3D one on board)

Optical Encoders

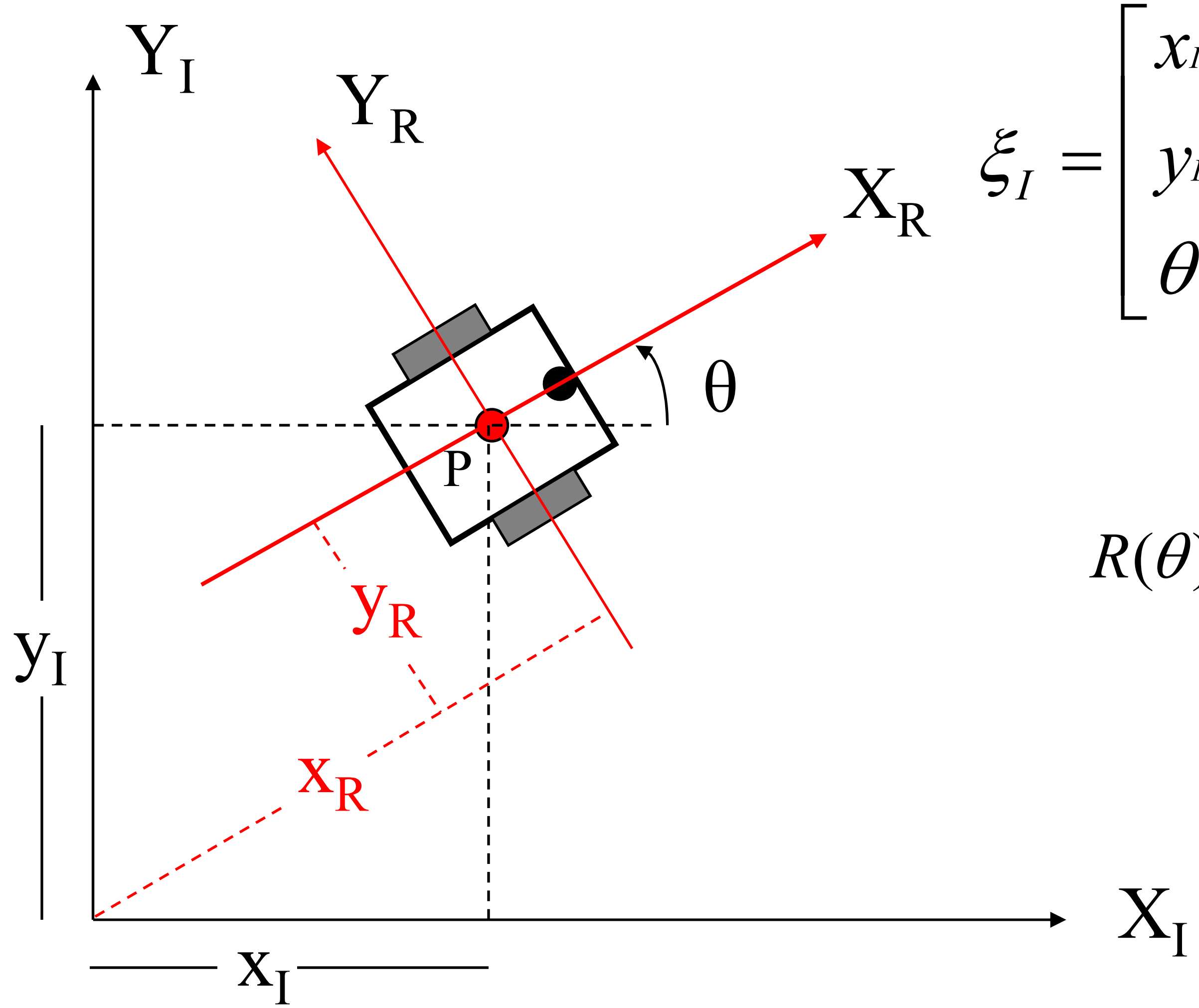
- Measure displacement (or speed) of the wheels
- Principle: mechanical light chopper consisting of photo-barriers (pair of light emitter and optical receiver) + pattern on a disc anchored to the motor shaft
- Quadrature encoder: 90º placement of 2 complete photo-barriers, 4x increase resolution + direction of movement
- Integrate wheel movements to get an estimate of the position -> odometry
- Typical resolutions: 64 - 2048 increments per revolution.
- For high resolution: interpolation





State	Ch A	Ch B
S_1	High	Low
S_2	High	High
S_3	Low	High
S_4	Low	Low

Pose (Position and Orientation) of a Differential-Drive Robot

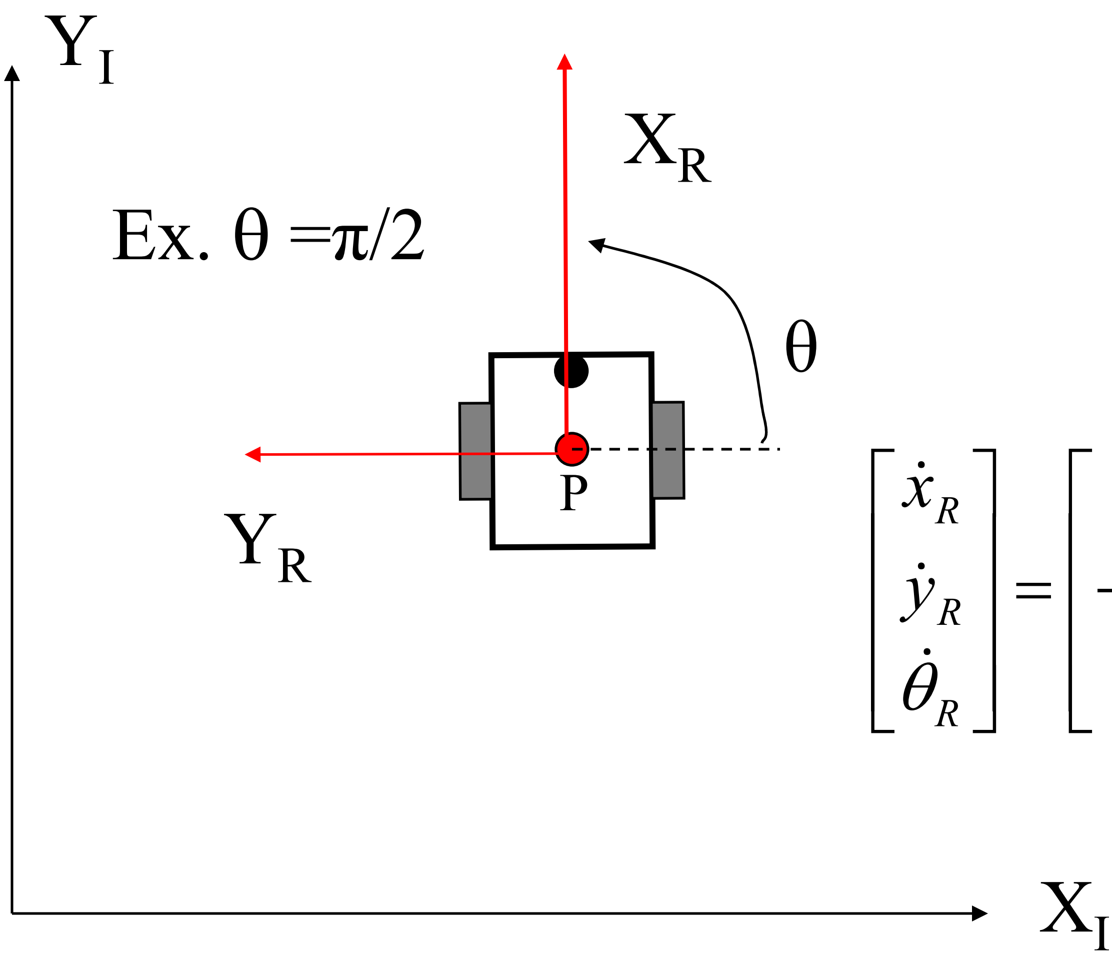


$$\xi_I = \begin{bmatrix} x_I \\ y_I \\ \theta \end{bmatrix} \quad \xi_R = \begin{bmatrix} x_R \\ y_R \\ \theta \end{bmatrix} = R(\theta)\xi_I$$

$$R(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

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

Absolute and Relative Pose of a Differential-Drive Robot



$$\dot{\xi}_R = R(\theta)\dot{\xi}_I$$

$$\begin{bmatrix} \dot{x}_R \\ \dot{y}_R \\ \dot{\theta}_R \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x}_I \\ \dot{y}_I \\ \dot{\theta}_I \end{bmatrix} = \begin{bmatrix} \dot{y}_I \\ -\dot{x}_I \\ \dot{\theta}_I \end{bmatrix}$$

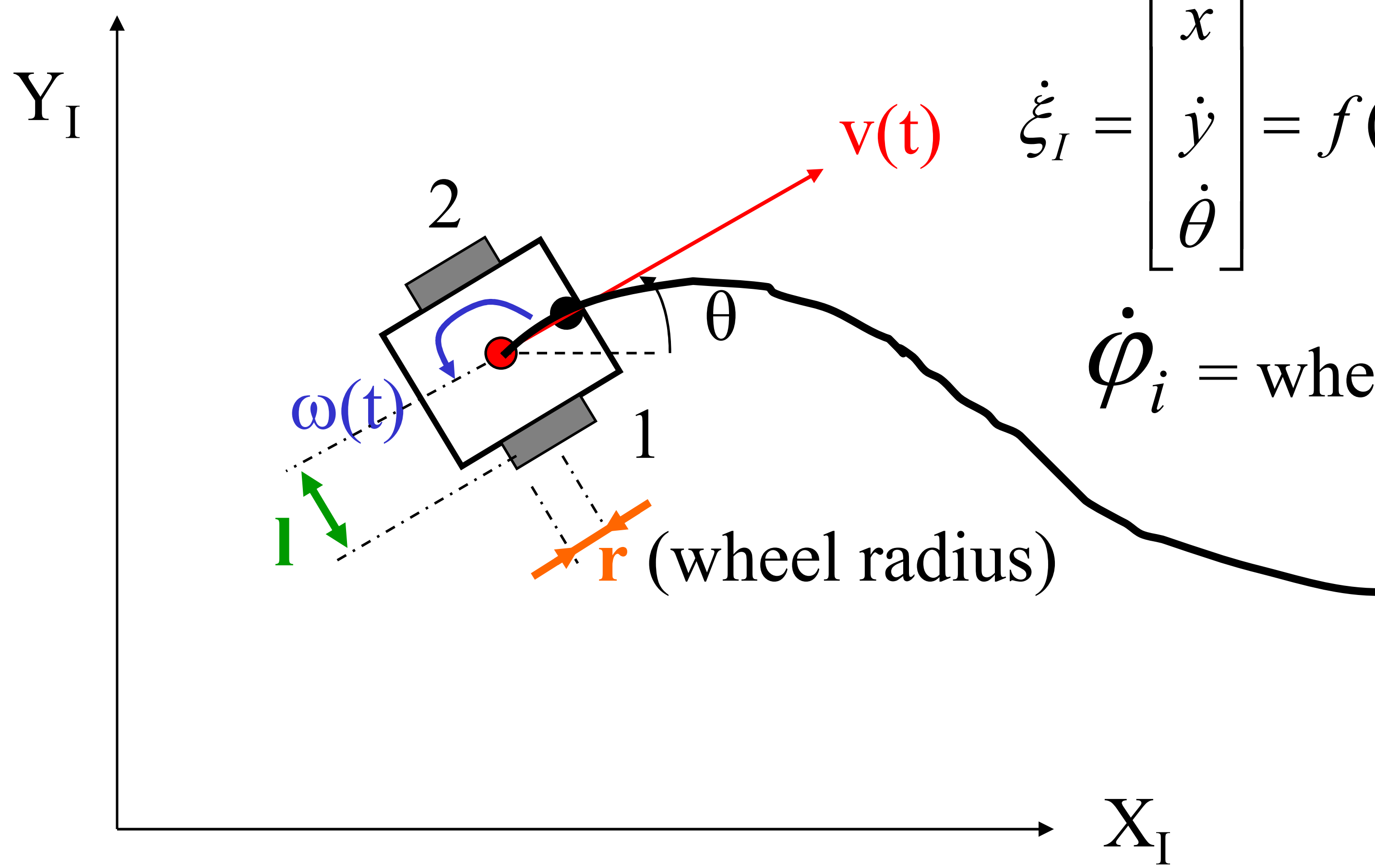
Forward Kinematic Model

How does the robot move given the wheel speeds and geometry?

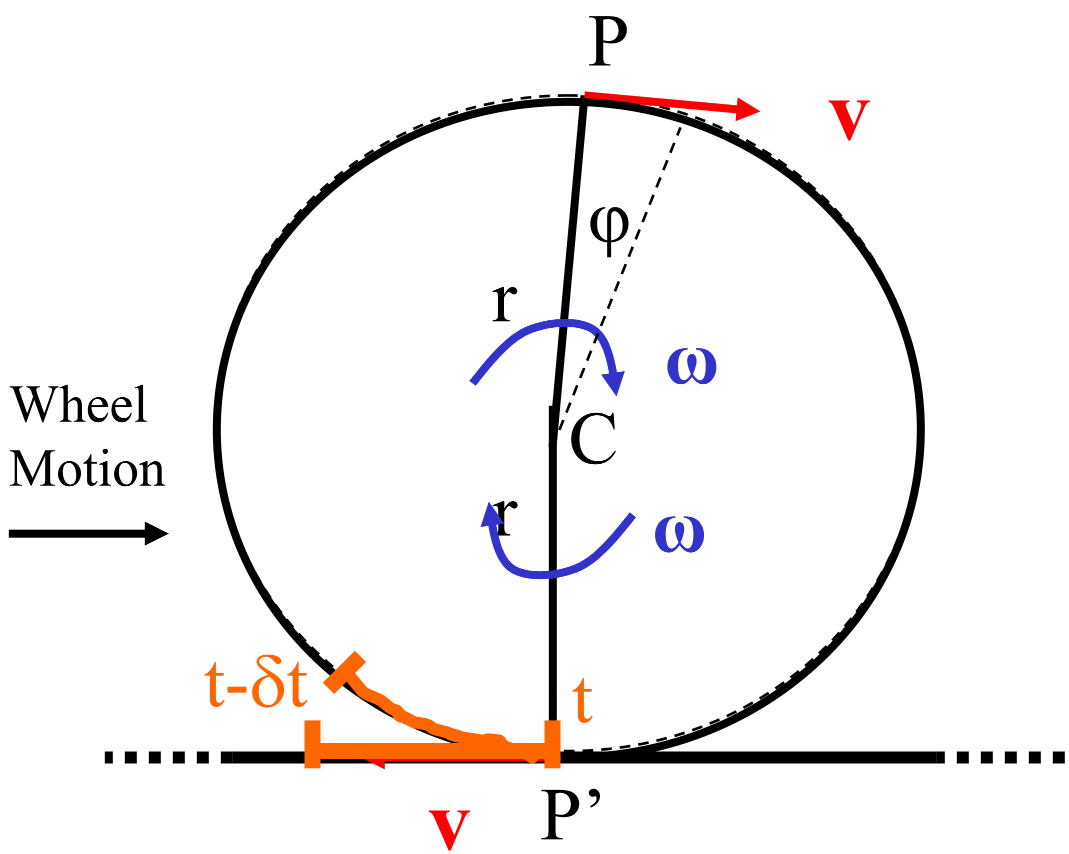
- Assumption: no wheel slip (rolling mode only)!
- In miniature robots no major dynamic effects due to low mass

$$\dot{\xi}_I = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = f(l, r, \theta, \dot{\varphi}_1, \dot{\varphi}_2)$$

$$\dot{\varphi}_i = \text{wheel i speed}$$



Recap ME/PHY Fundamentals



$$\boxed{v = \omega r = \dot{\varphi} r}$$

v = tangential speed

ω = rotational speed

r = rotation radius

φ = rotation angle

C = rotation center

P = peripheral point

P'= contact point at time t

Rolling!

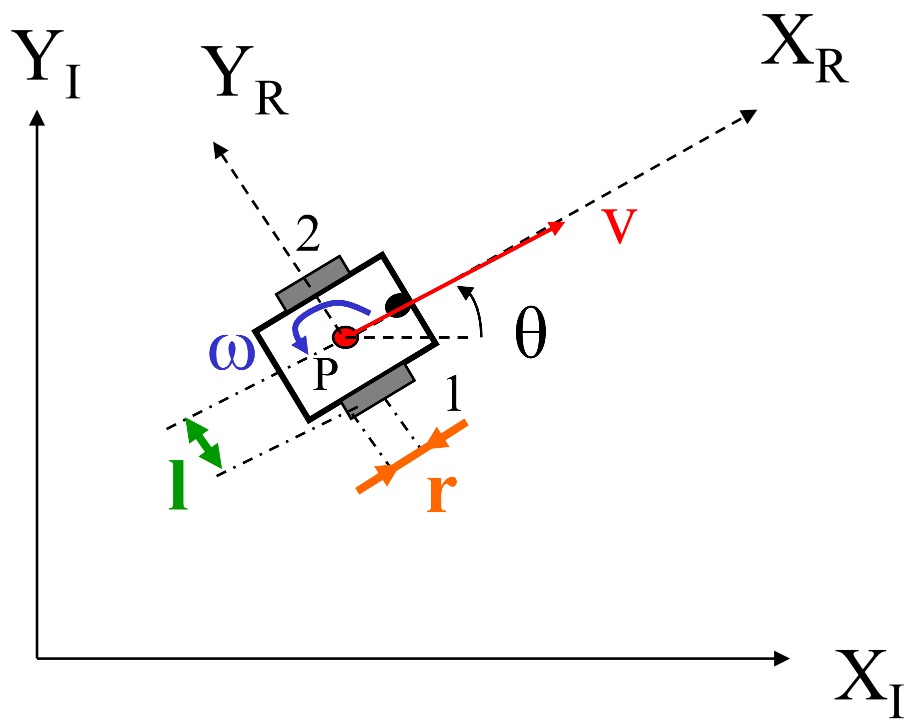
Forward Kinematic Model

Linear speed = average wheel speed 1 and 2:

$$v = \frac{r\dot{\varphi}_1}{2} + \frac{r\dot{\varphi}_2}{2}$$

Rotational speed = sum of rotation speeds (wheel 1 clockwise, wheel 2 counter-clockwise):

$$\omega = \frac{r\dot{\varphi}_1}{2l} + \frac{-r\dot{\varphi}_2}{2l}$$



Idea: linear superposition of individual wheel contributions

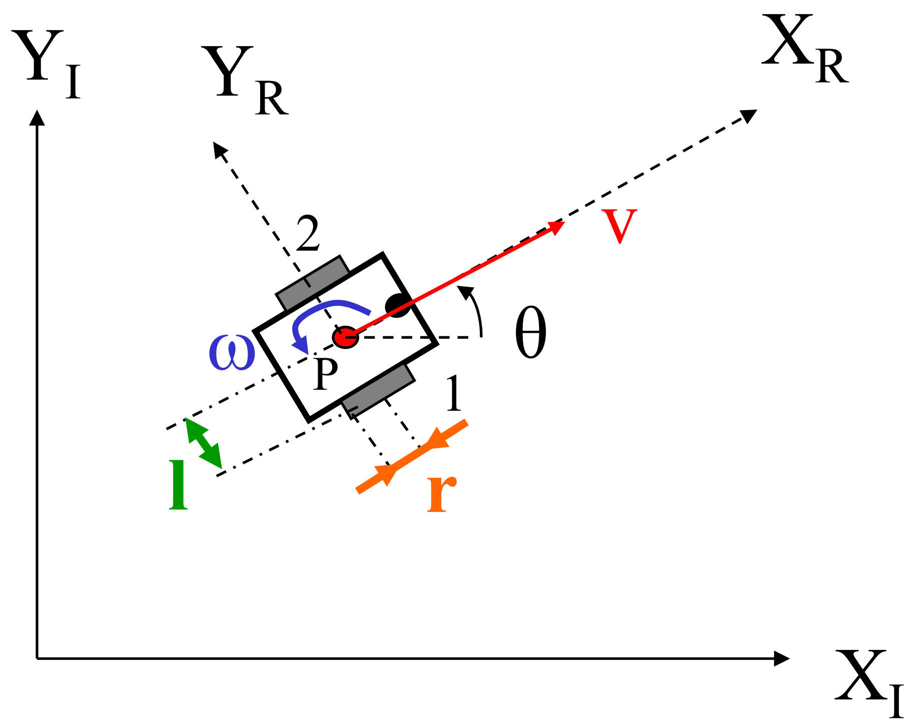
Forward Kinematic Model

1. $\dot{\xi}_I = R^{-1}(\theta)\dot{\xi}_R$

2. $\dot{x}_R = v = \dfrac{r\dot{\varphi}_1}{2} + \dfrac{r\dot{\varphi}_2}{2}$

3. $\dot{y}_R = 0$

4. $\dot{\theta}_R = \omega = \dfrac{r\dot{\varphi}_1}{2l} + \dfrac{-r\dot{\varphi}_2}{2l}$



$$\dot{\xi}_I = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dfrac{r\dot{\varphi}_1}{2} + \dfrac{r\dot{\varphi}_2}{2} \\ 0 \\ \dfrac{r\dot{\varphi}_1}{2l} + \dfrac{-r\dot{\varphi}_2}{2l} \end{bmatrix}$$

Odometry

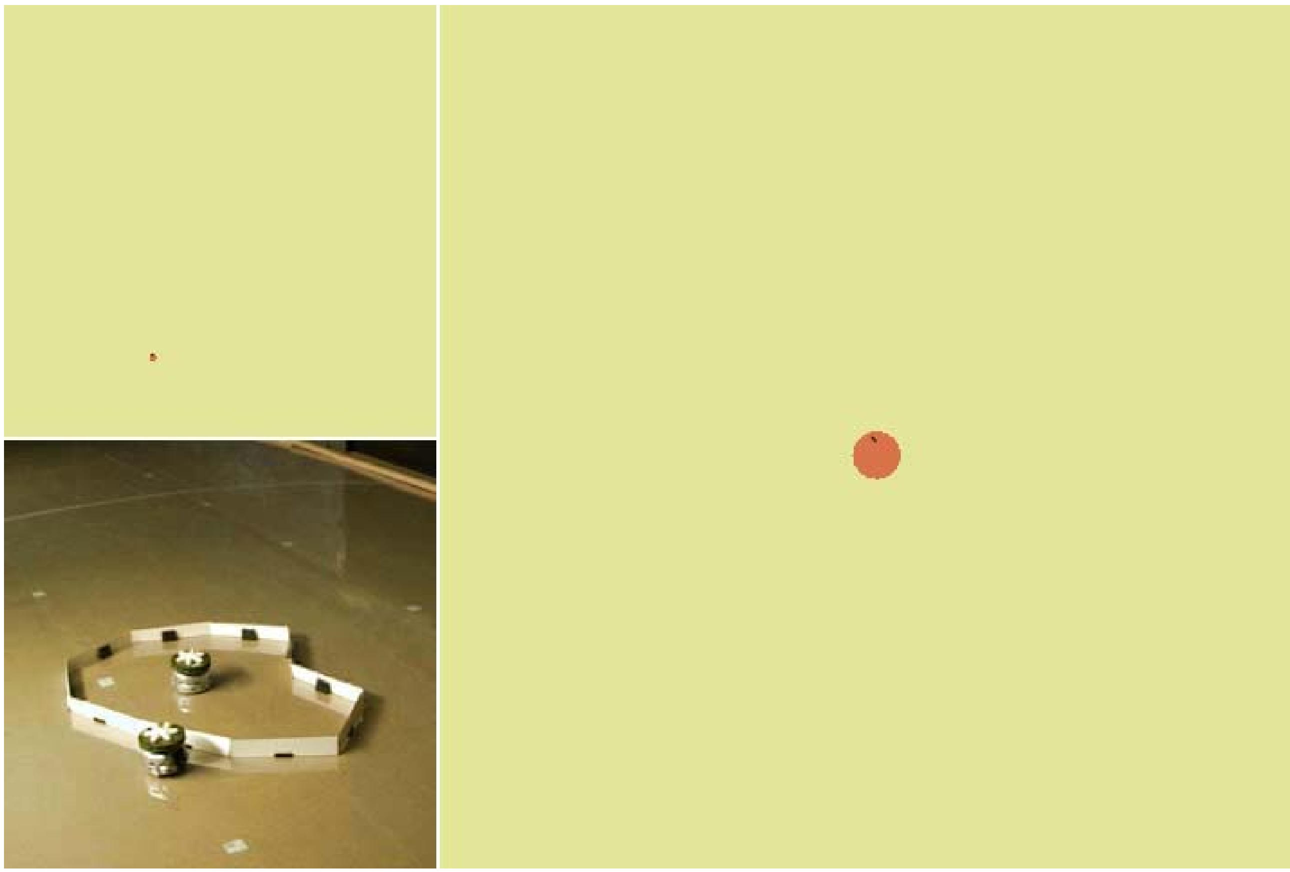
- Q: given our absolute pose over time, how can we calculate the robot pose after some time *t*?
- A: integrate!
- Given the kinematic forward model, and assuming no slip on both wheels

$$\xi_I(T) = \xi_{I0} + \int_0^T \dot{\xi}_I dt = \xi_{I0} + \int_0^T R^{-1}(\theta)\dot{\xi}_R dt$$

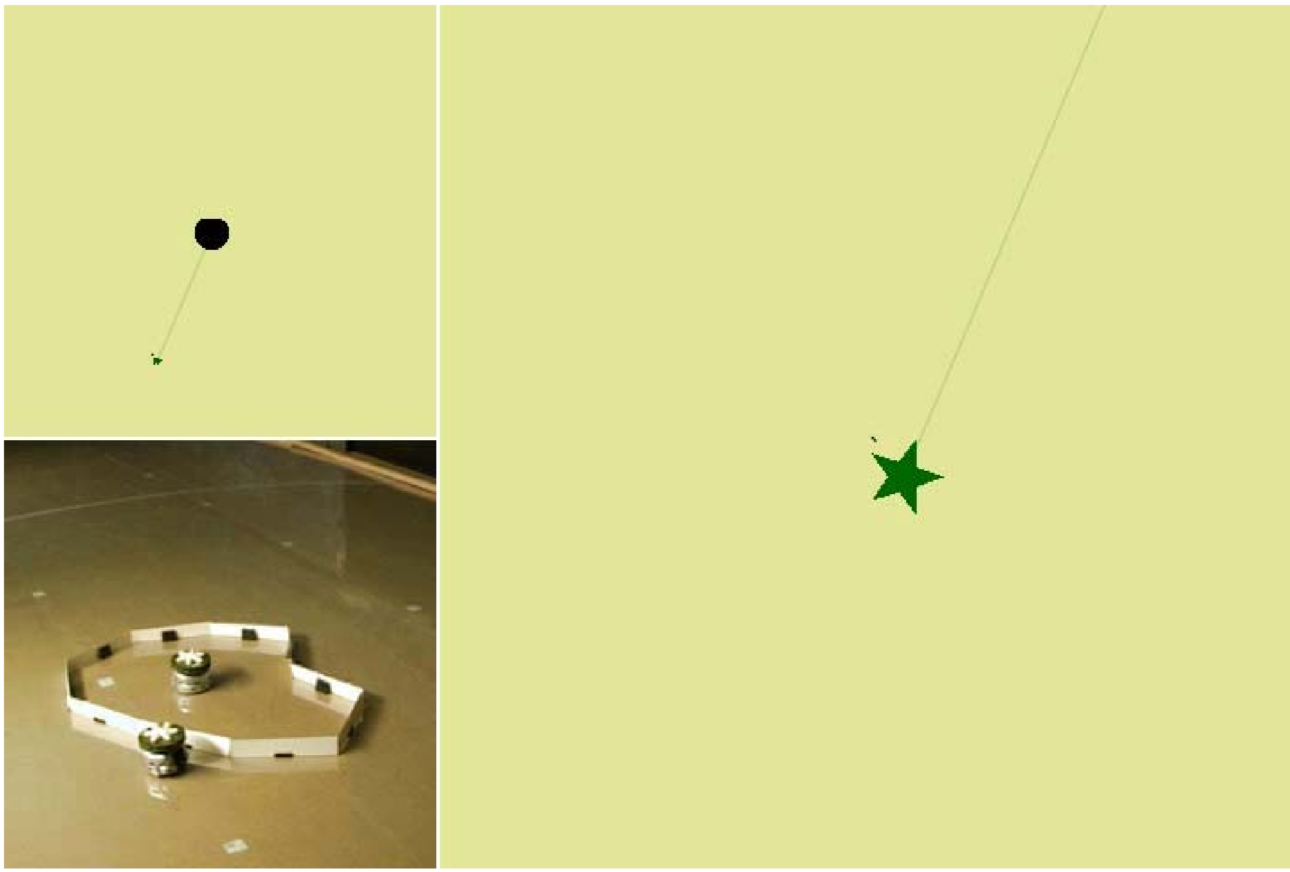
- Given an initial pose ξ_{I0}, after time T, the pose of the vehicle will be $\xi_I(T)$
- $\xi_I(T)$ computable with wheel speed 1, wheel speed 2, and parameters r and l
- Note: in practice wheel slippage always present → pose error based on odometry is cumulative and incrementally increases; see later for handling this error

Examples of Positioning Systems in Action

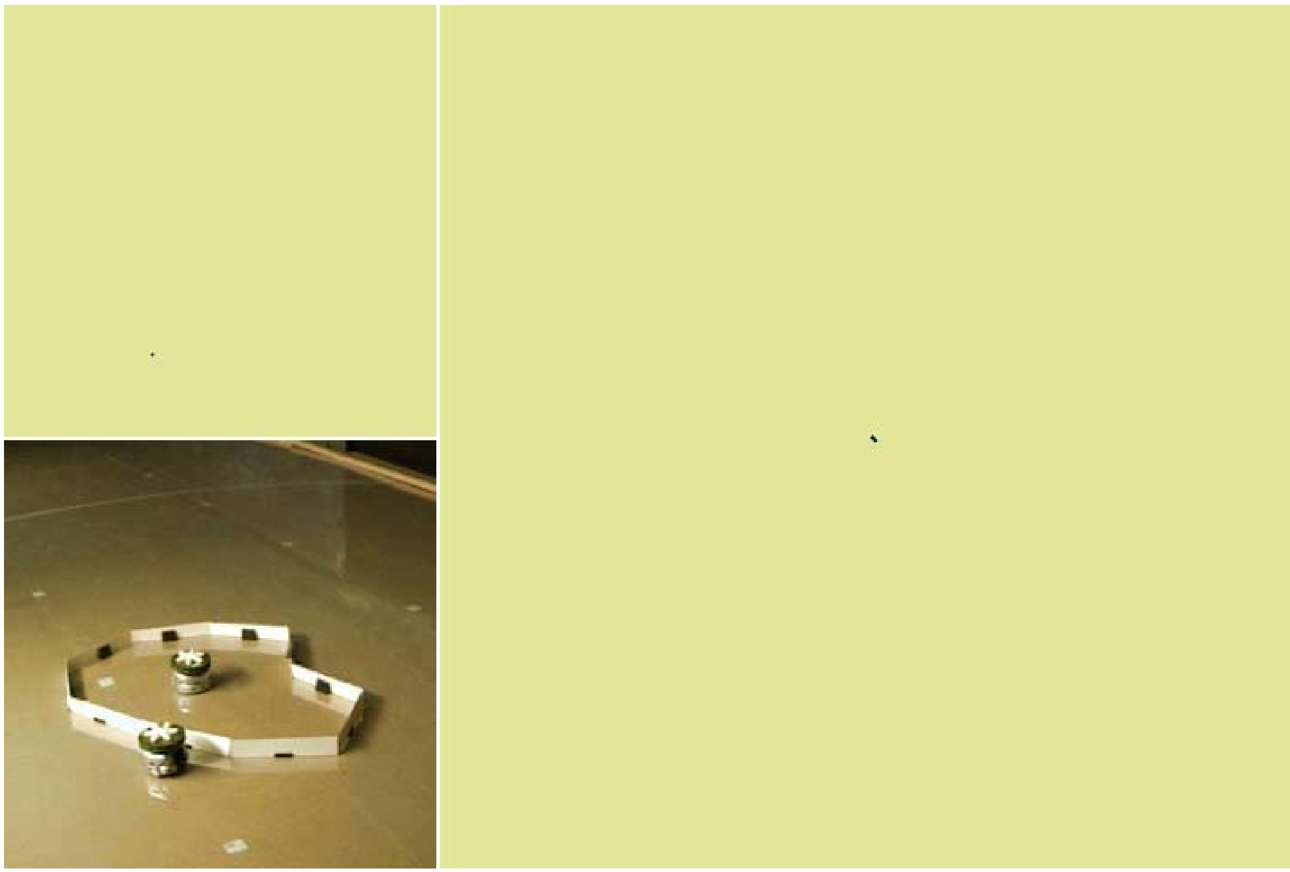
Overhead Camera System



Range & Bearing IR+RF System



Odometry



Robot Localization with Uncertainties: Sources and Handling Methods

Outline

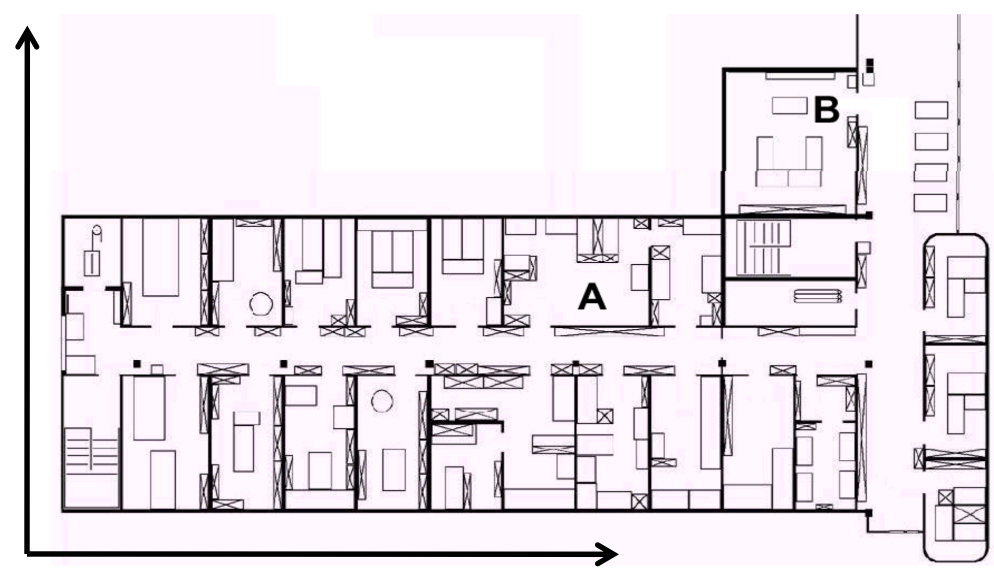
- Sensors for localization
- Odometry-based navigation
- Belief representation part 1
- Feature-based navigation
- Belief representation part 2

Robot Localization

- Key task for:
  - Path planning
  - Mapping
  - Referencing
  - Coordination

- Type of localization
  - Absolute coordinates
  - Local coordinates
  - Topological information

N 46° 31’ 13’’
E 6 ° 34’ 04’’



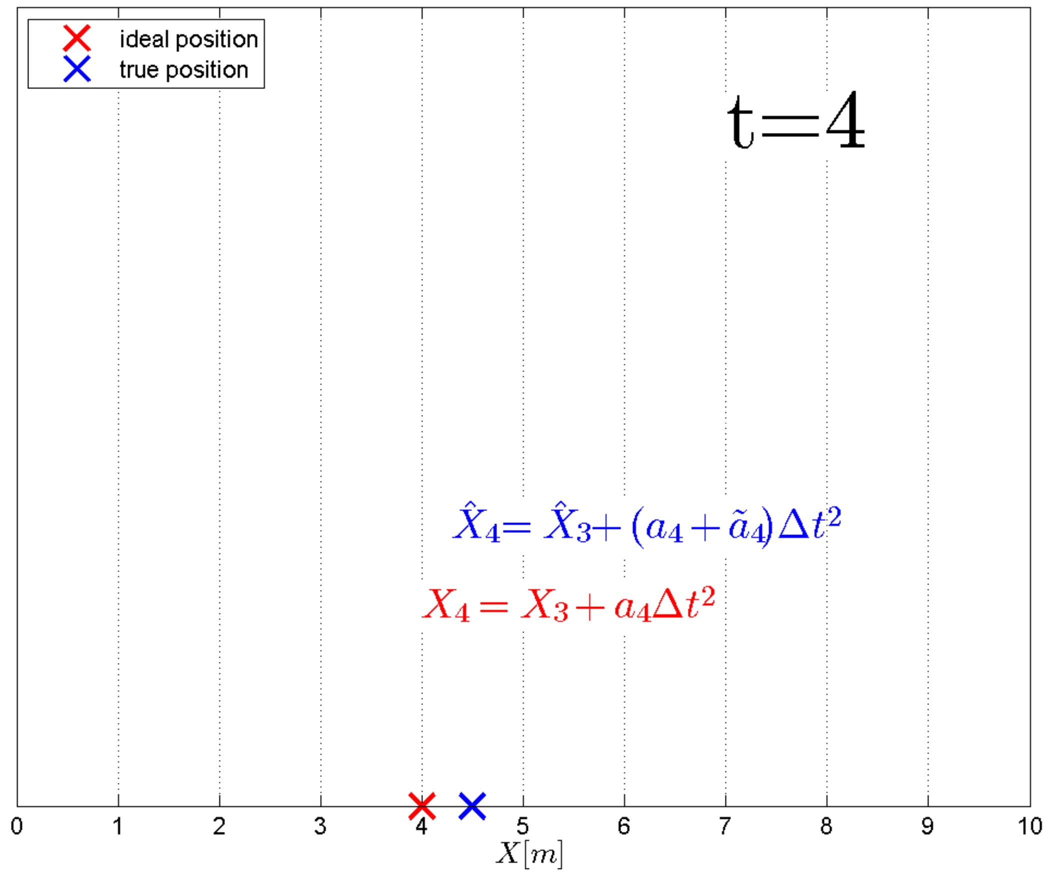
Sensors for localization

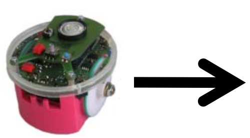
- Proprioceptive sensors:
  - Epuck:
    - 3D accelerometer
    - Motor step counter
  - Others:
    - Wheel encoder
    - Odometer
- Exteroceptive sensors:
  - Epuck:
    - IR range proximity sensor
    - Camera
  - Others:
    - Laser range finder
    - Ultrasonic range finder

Sensors for localization

- Proprioceptive sensors:
    - Epuck:
        - 3D accelerometer
        - Motor step counter
    - Others:
        - Wheel encoder
        - Odometer
- Exteroceptive sensors:
    - Epuck:
        - IR range proximity sensor
        - Camera
    - Others:
        - Laser range finder
        - Odometer

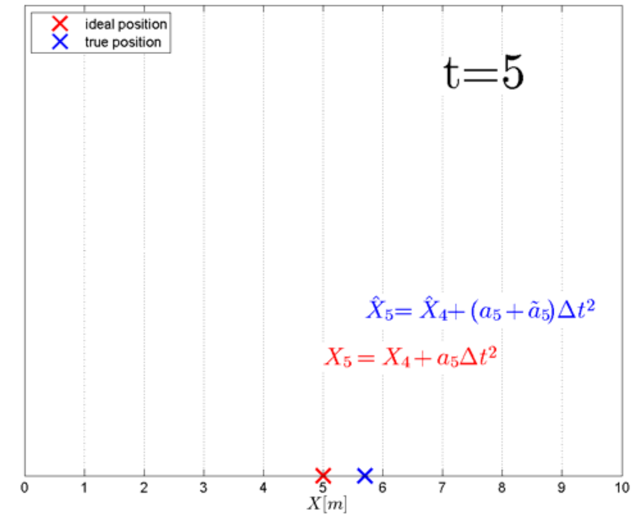
Accelerometer based odometry

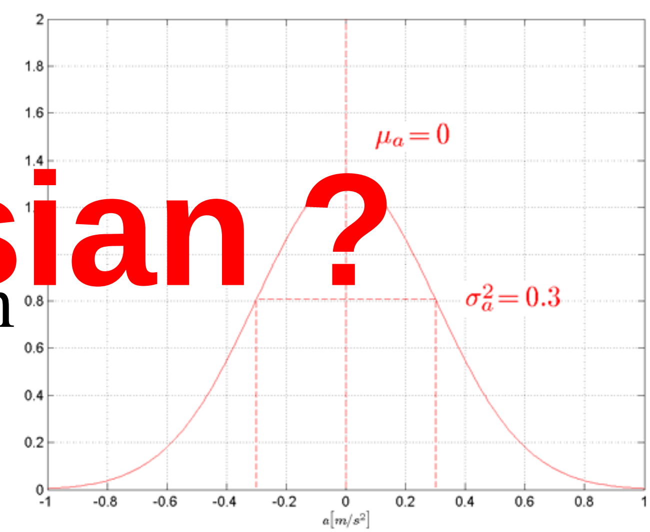




Error modeling

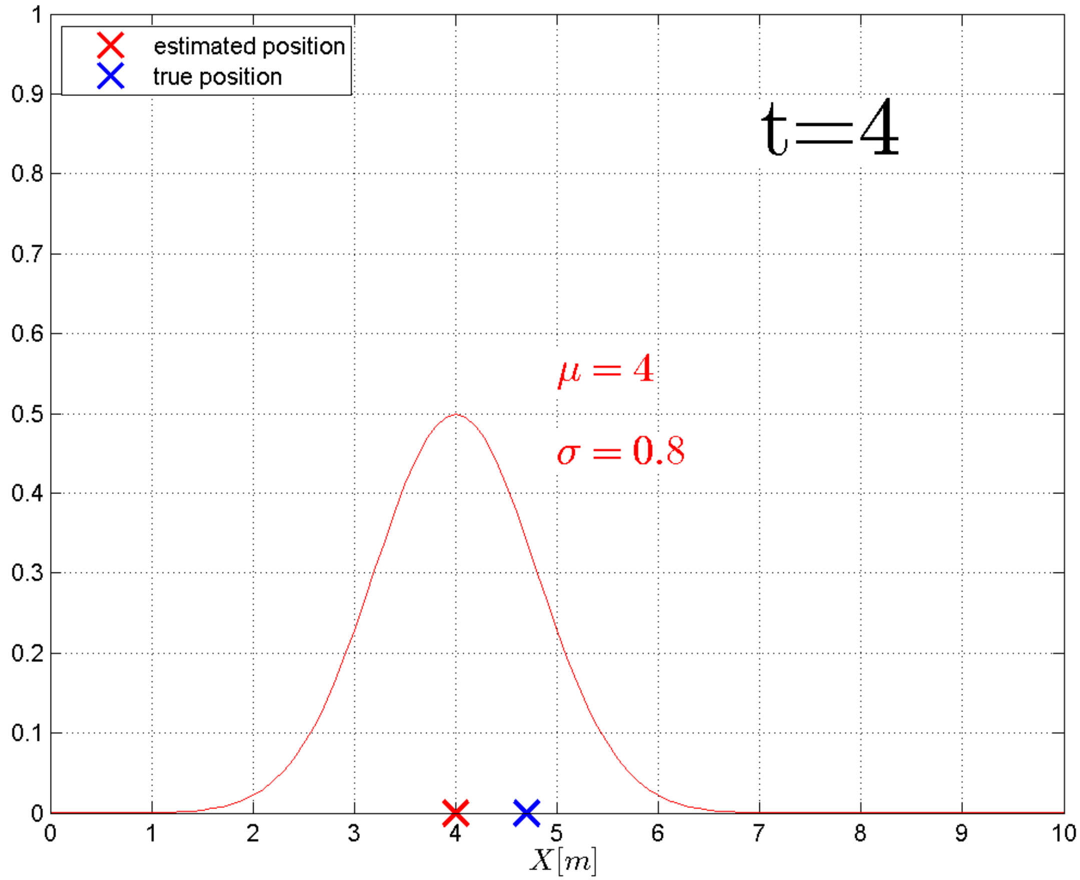
- Error happens!
- Odometry error is cumulative.
  → grows without bound
- We need to be aware of it.
  → We need to model odometry error.
  → We need to model sensor error.
- Acceleration is random variable A drawn from "mean-free" Gaussian ("Normal") distribution.
  → Position X is random variable with Gaussian distribution.

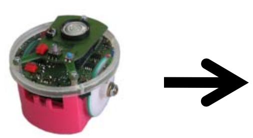




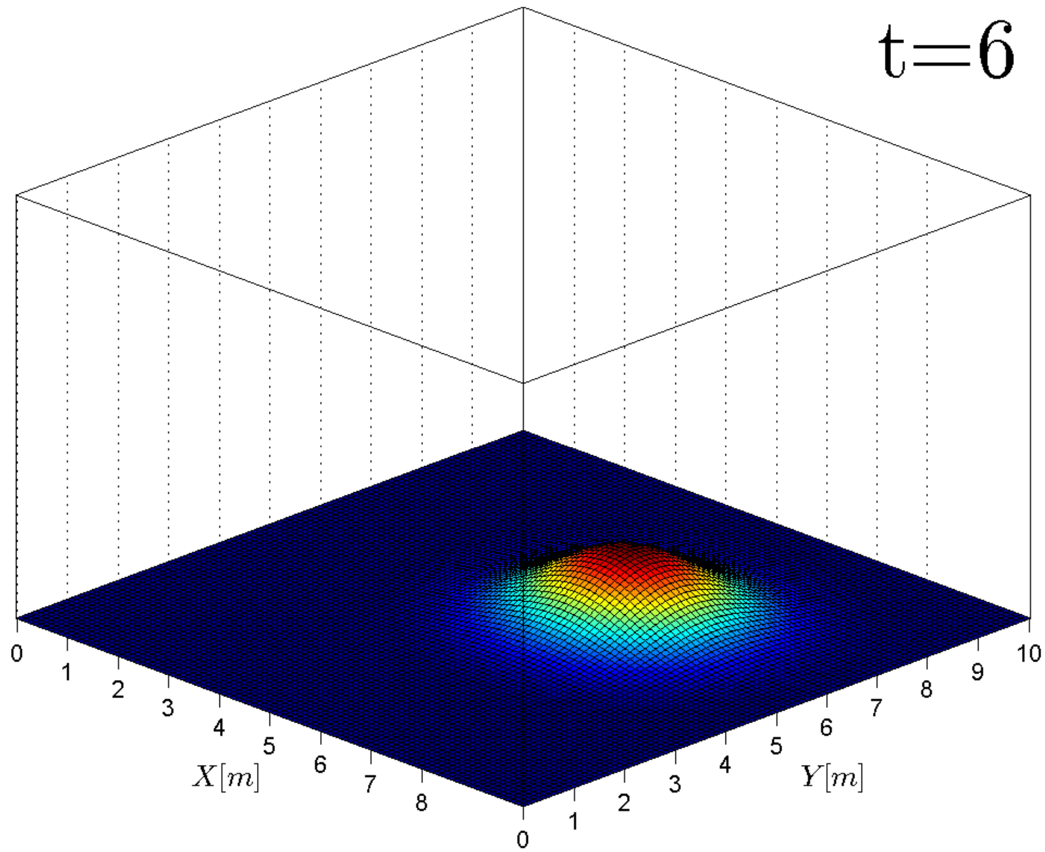
Why Gaussian ?

Accelerometer based odometry

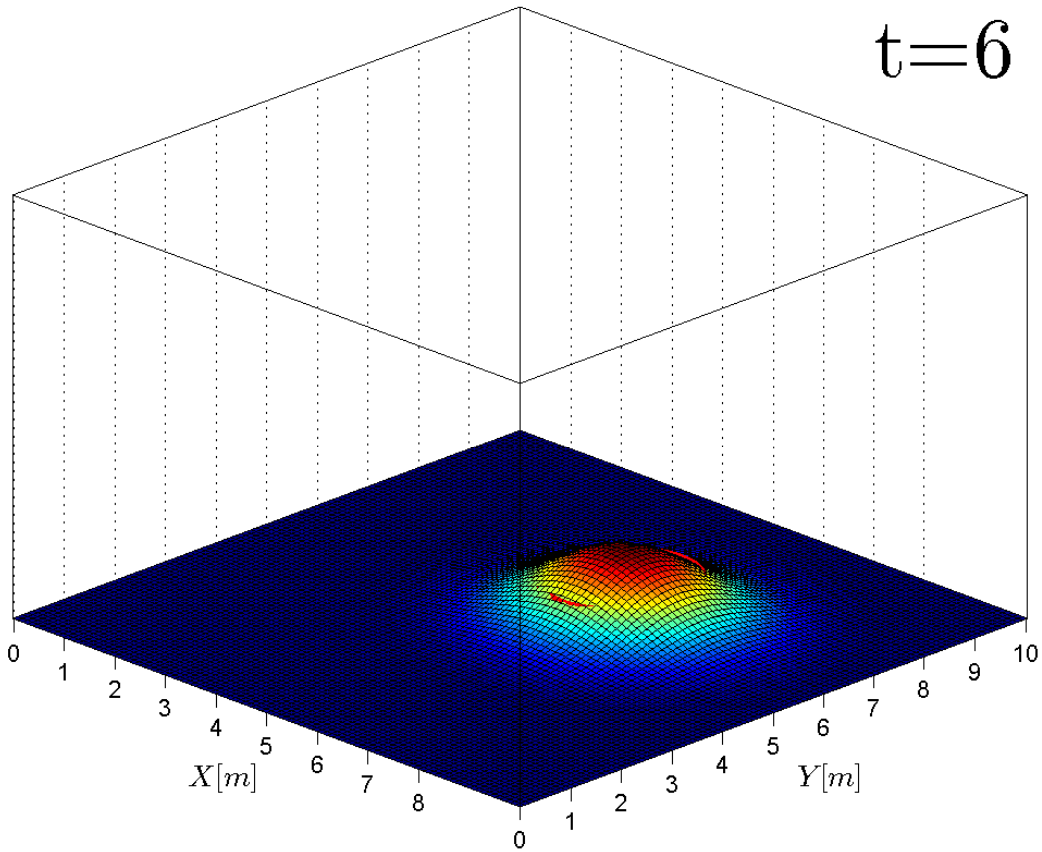




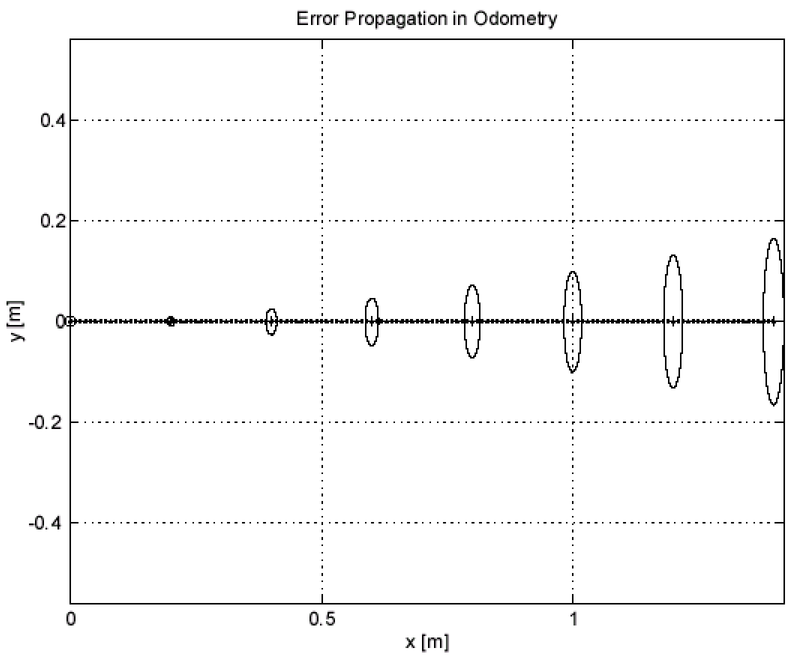
Accelerometer based odometry 2D

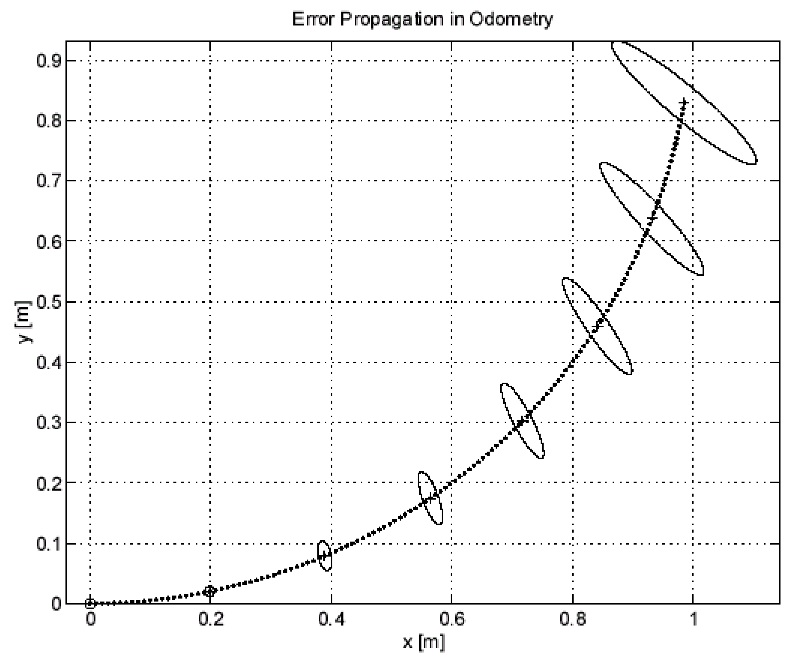


Accelerometer based odometry 2D



Classical 2D representation



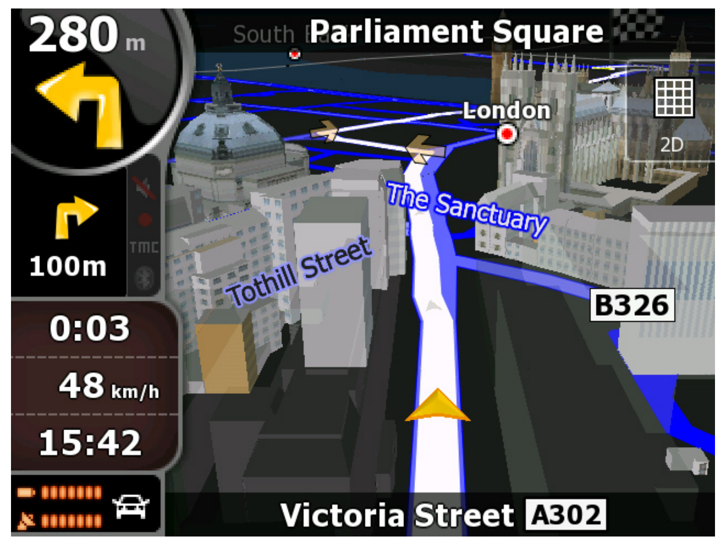


Courtesy of R. Siegwart and R. Nourbakhsh

Real world odometry examples

- Human in the dark
  - Very **bad** odometry sensors
  - $d_{Odometry} = O(m)$
- (Nuclear) Submarine
  - Very **good** odometry sensors
  - $d_{Odometry} = O(10^3\ km)$
- Navigation system in tunnel uses dead reckoning based on
  - Last velocity as measured by GPS
  - Car's odometer, compass

Courtesy of US Navy



Courtesy of NavNGo

Features

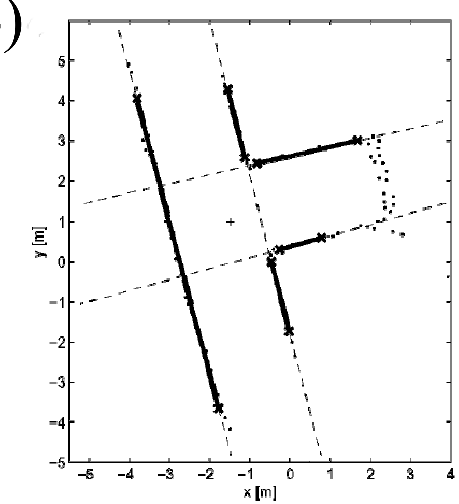
- Odometry based position error grows without bound.
- Use relative measurement to features ("landmarks") to reduce position uncertainty
- ***Feature***:
  - Uniquely identifiable
  - Position is known
  - We can obtain relative measurements between robot and feature (usually angle or range).
- Examples:
  - Doors, walls, corners, hand rails
  - Buildings, trees, lanes
  - GPS satellites

Courtesy of Albert Huang

Automatic feature extraction

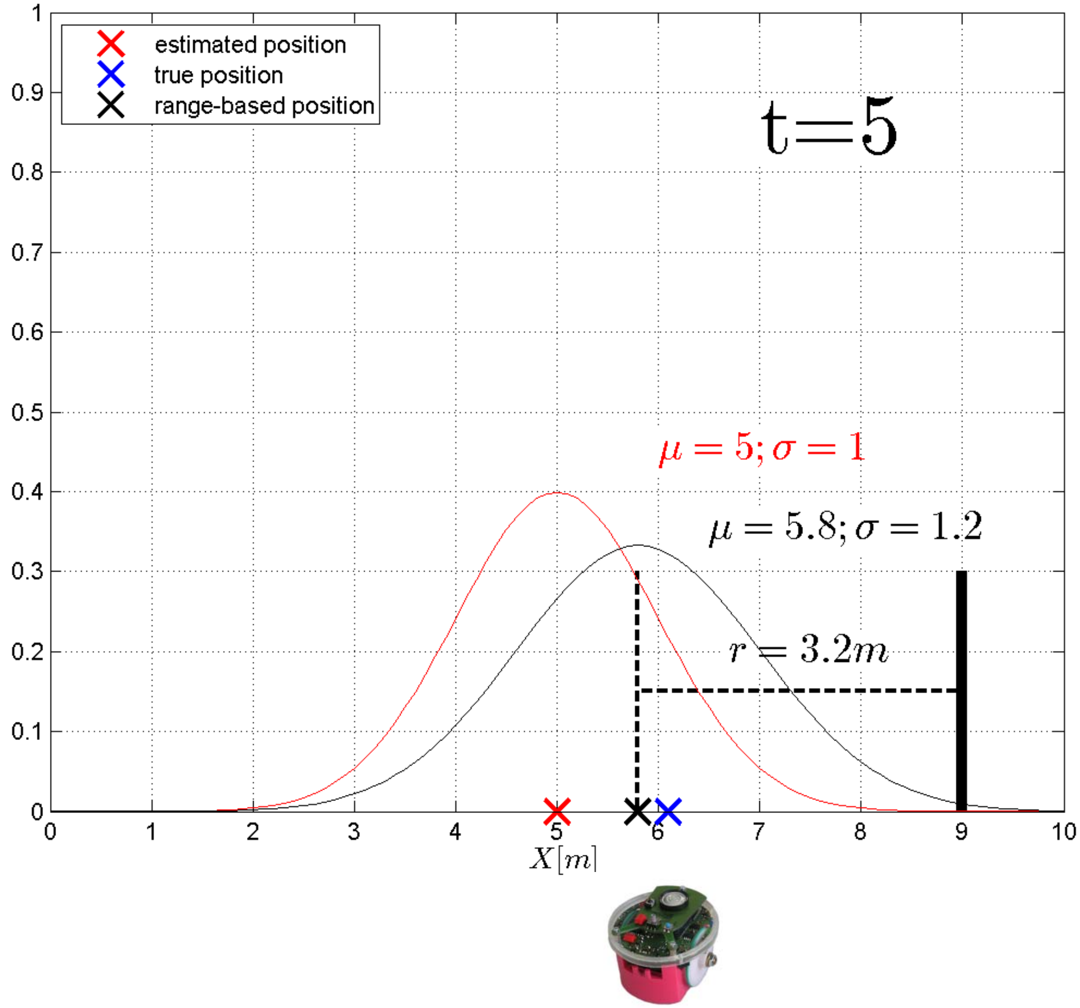
- High level features:
  - Doors, persons
- Simple visual features:
  - Edges (Canny Edge Detec
  - Corner (Harris Corner De 8)
  - ***S****cale* ***I****nvariant* ***F****eature* ***T*** tion (2004)
- Simple geometric feature
  - Lines
  - Corners
- “Binary” feature

Complexity



Courtesy of R. Siegwart and R. Nourbakhsh

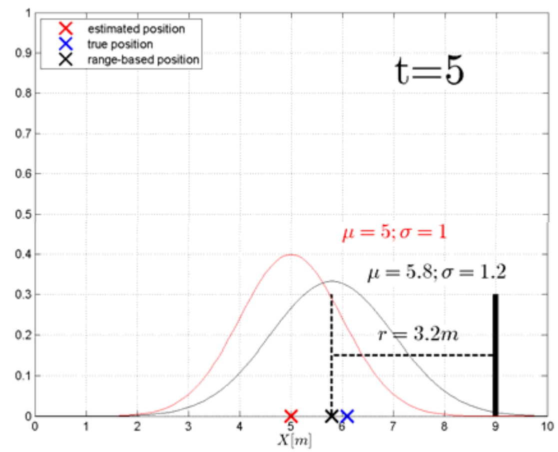
Feature-based navigation



Sensor fusion

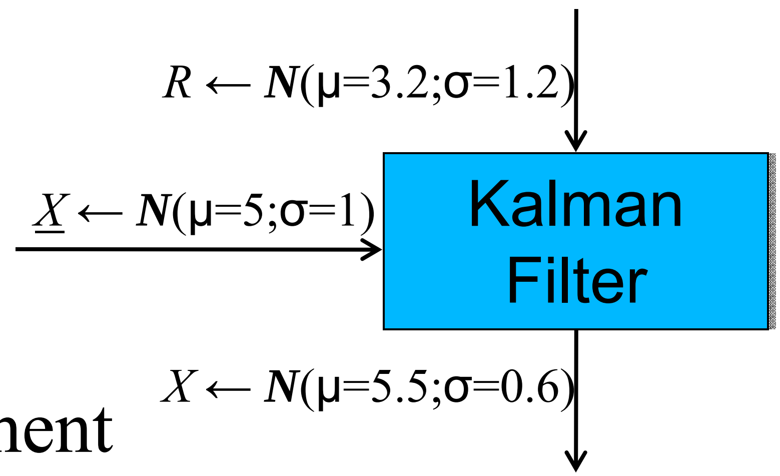
- Given:
  - Position estimate $\underline{X} \leftarrow \boldsymbol{N}(\mu=5;\sigma=1)$
  - Range estimate $R \leftarrow \boldsymbol{N}(\mu=3.2;\sigma=1.2)$

    What is the best estimate AFTER incorporating r ?



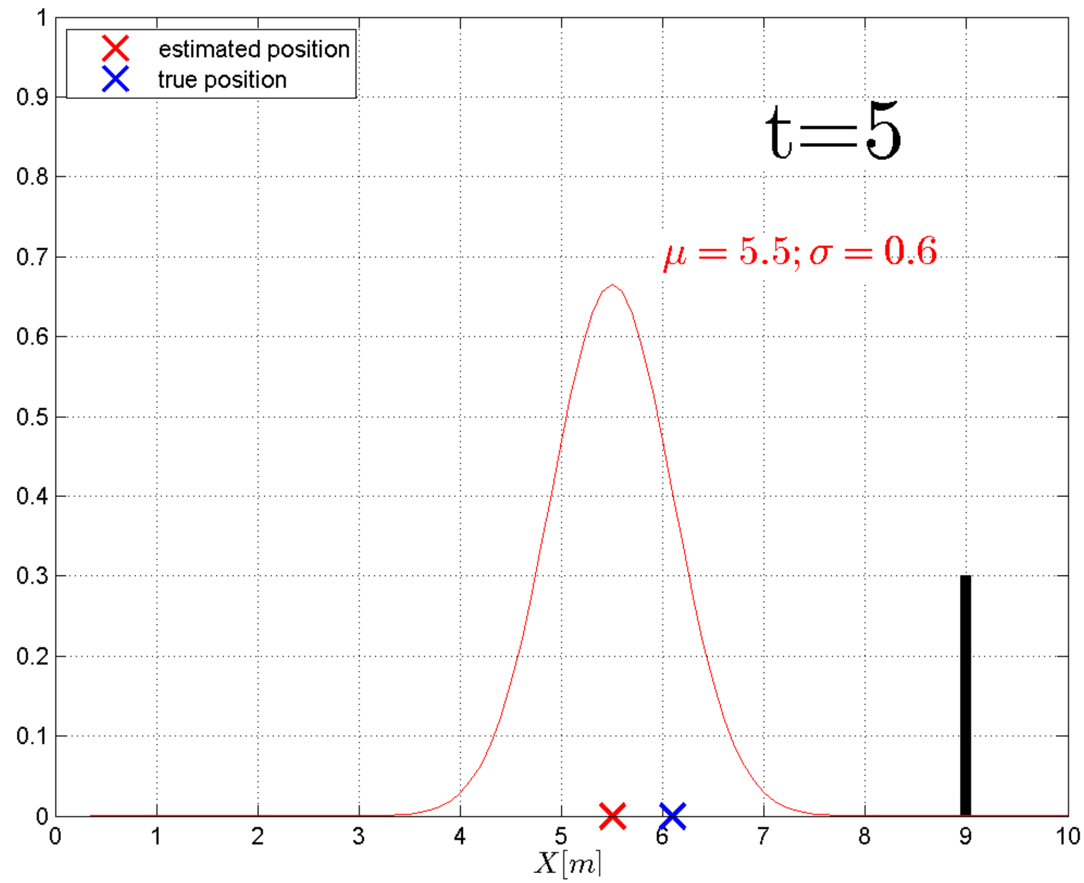
→ ***Kalman Filter***

- Requires:
  - Gaussian noise distribution for all measurements
  - Linear motion and measurement model
  - …



[cs.unc.edu/~welch/media/pdf/**maybeck**_ch1.pdf]

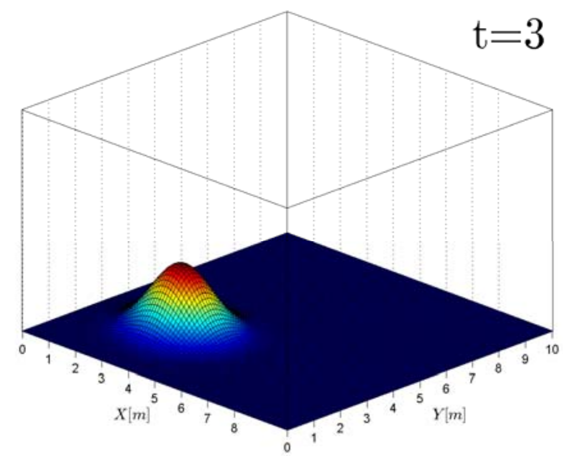
Feature-based navigation

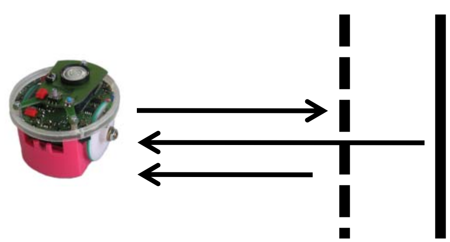


Feature-based navigation

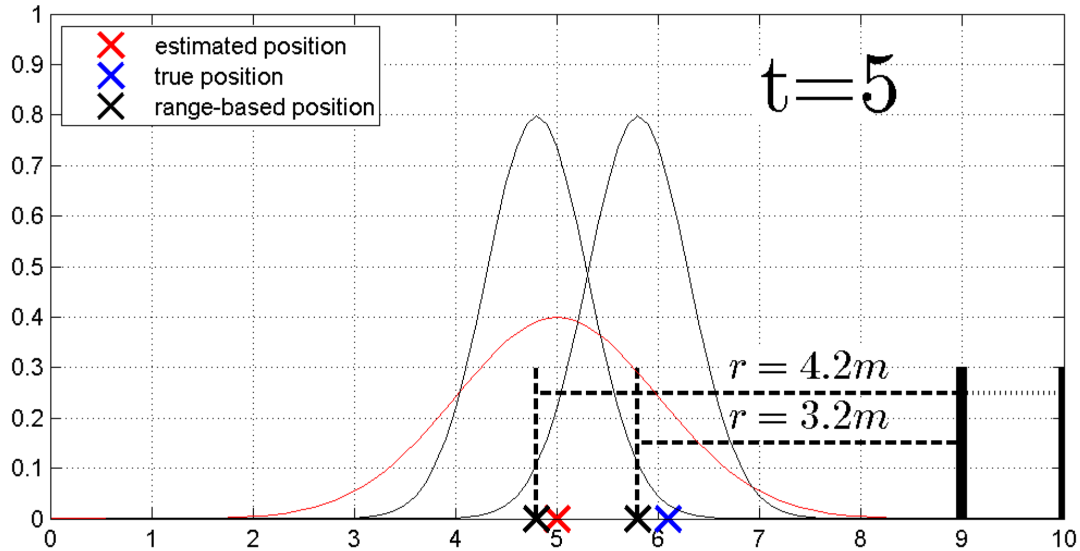
Belief representation trough Gaussian distribution

- Advantages:
  - Compact (only mean and variance required)
  - Continuous
  - Powerful tools (Kalman Filter)
- Disadvantages:
  - Requires Gaussian noise assumption
  - Uni-modal
  - Cannot represent ignorance ("kidnapped robot problem")

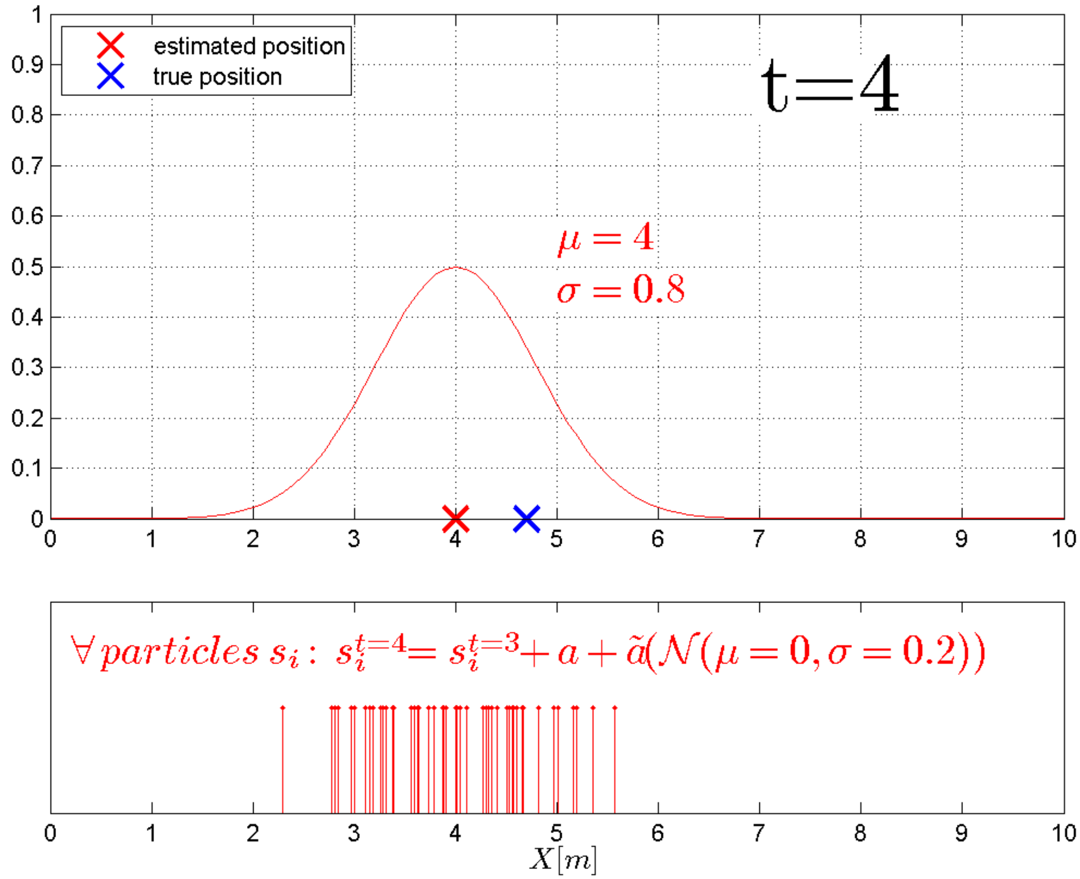




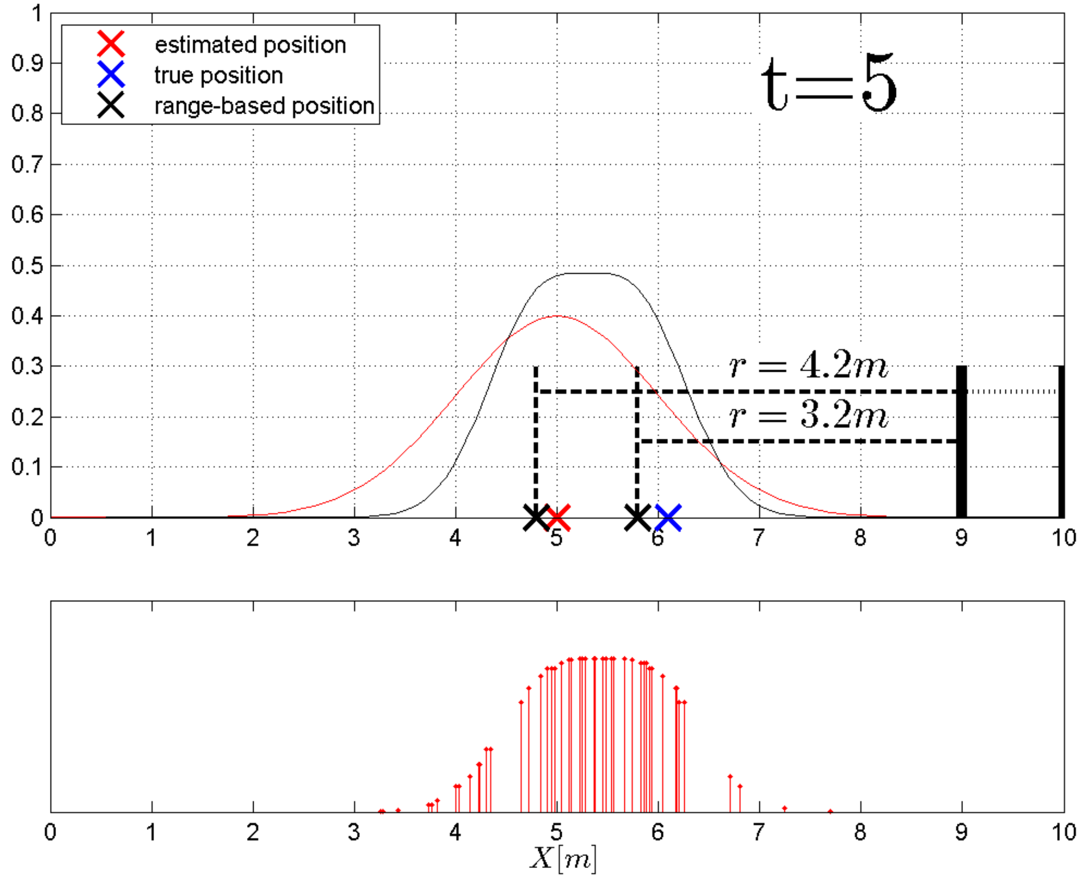
Feature-based navigation



Feature-based navigation



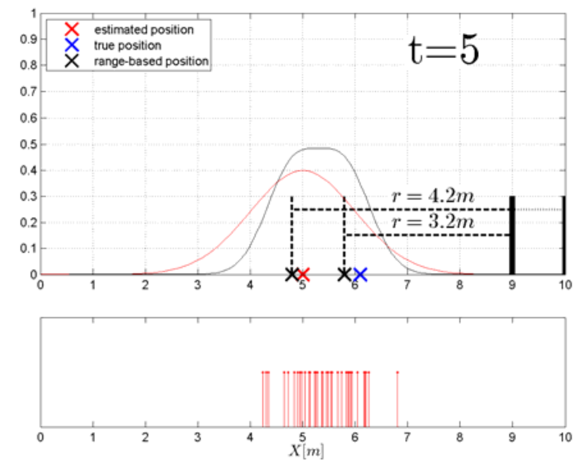
Feature-based navigation



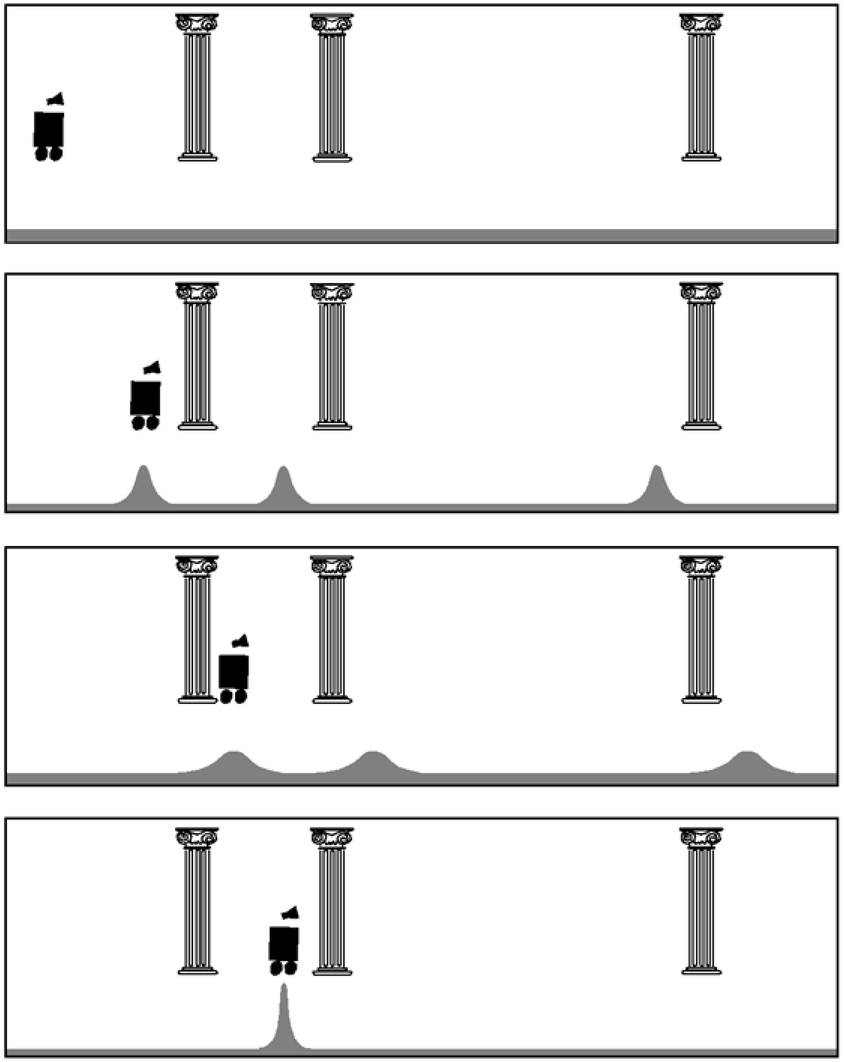
Feature-based navigation

Belief representation trough particle distribution

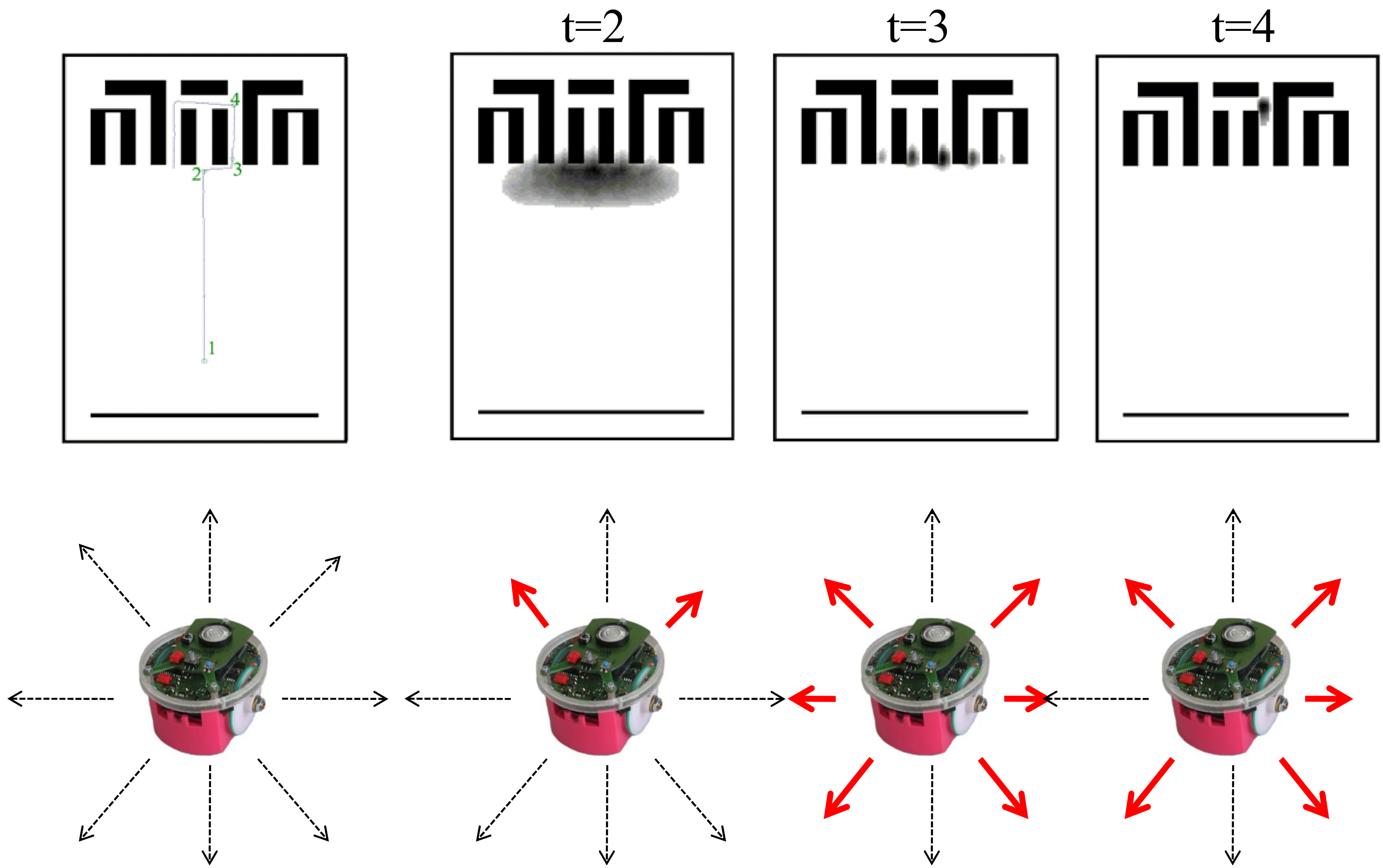
- Advantages:
  - Can model arbitrary beliefs
  - No assumptions on noise characteristic
- Disadvantages:
  - No unique solution
  - Not continuous
  - Computationally expensive
  - Tuning required



Feature-based navigation



Feature-based navigation



Error propagation in Wheel-Based Odometry

Sensor noise → Position noise

- Until now: used acceleration sensor

$\tilde{a} \to \tilde{x} = \tilde{a} t^2$

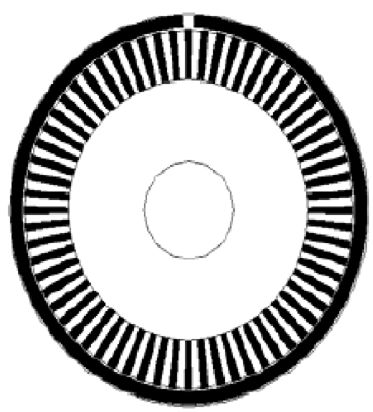
$\sigma_a \to \sigma_x$???

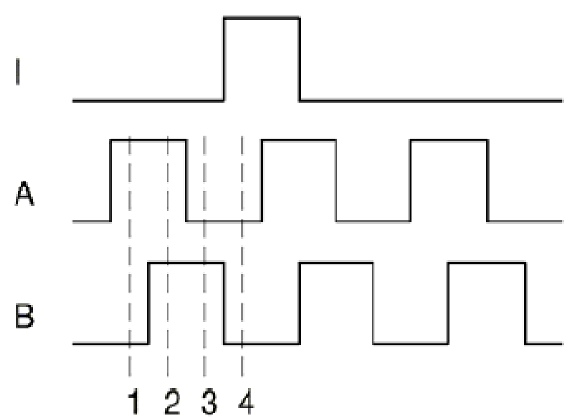
- Now use: wheel encoder

$c_l d \to \Delta s_l$

$c_r d \to \Delta s_r$

$\sigma_{\Delta s_l}, \sigma_{\Delta s_r} \to \sigma_x, \sigma_y, \sigma_\theta$???





State	Ch A	Ch B
S_1	High	Low
S_2	High	High
S_3	Low	High
S_4	Low	Low

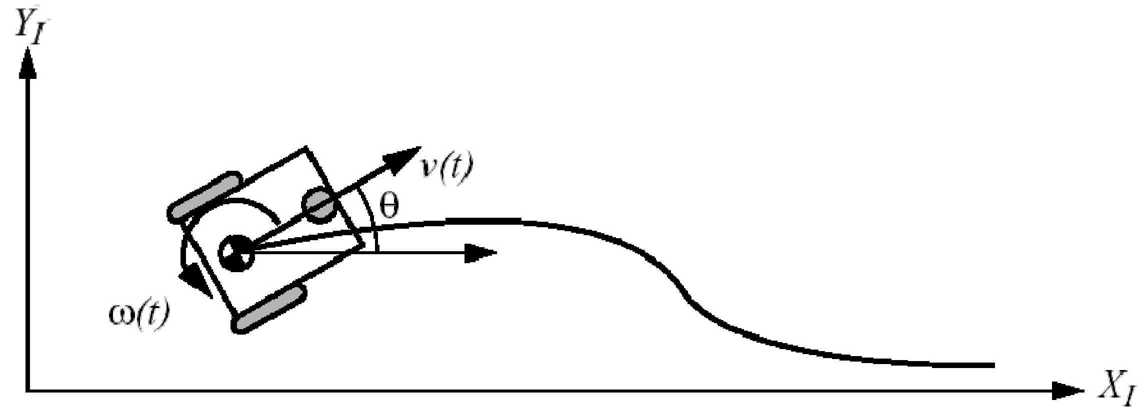
Sensor → Position

$$\Delta s = \frac{\Delta s_r + \Delta s_l}{2}$$

$$\Delta x = \Delta s \cos(\theta + \frac{\Delta\theta}{2})$$

$$\Delta y = \Delta s \sin(\theta + \frac{\Delta\theta}{2})$$

$$\Delta\theta = \frac{\Delta s_r + \Delta s_l}{b}$$



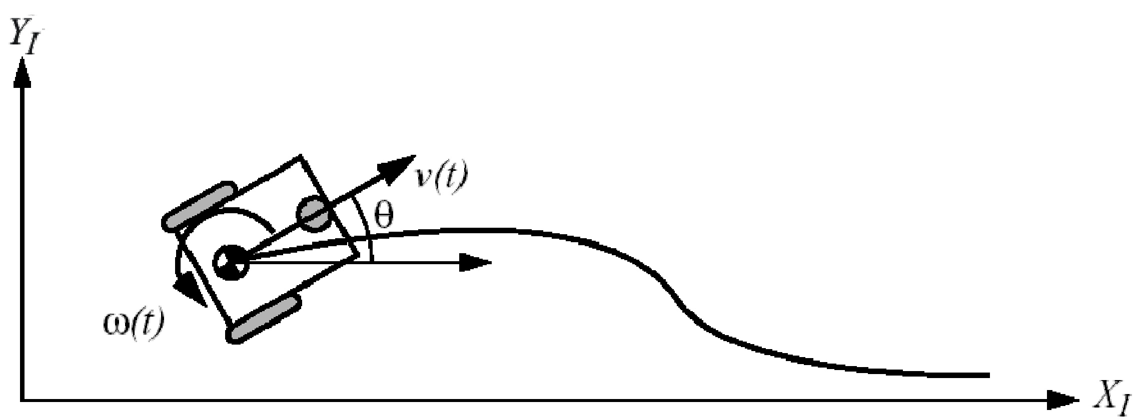
$$p = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \xrightarrow{t' = t + \Delta t} p' = \begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix}$$

$$p' = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} \Delta s \cos(\theta + \Delta\theta / 2) \\ \Delta s \sin(\theta + \Delta\theta / 2) \\ \Delta\theta \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} \frac{\Delta s_r + \Delta s_l}{2} \cos(\theta + \frac{\Delta s_r + \Delta s_l}{2b}) \\ \frac{\Delta s_r + \Delta s_l}{2} \sin(\theta + \frac{\Delta s_r + \Delta s_l}{2b}) \\ \frac{\Delta s_r + \Delta s_l}{b} \end{bmatrix}$$

Sensor noise → Position noise

- Add noise
  - Errors are independent
  - Errors are independent of direction
  - Errors are proportional to the distance traveled

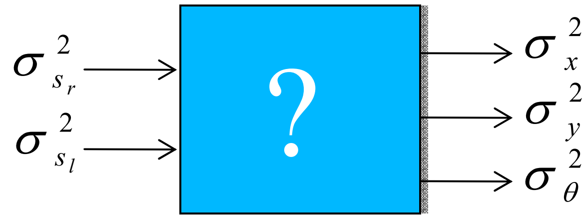
$$\Sigma_{\Delta} = \operatorname{cov}(\Delta s_r, \Delta s_l) = \begin{bmatrix} k_r|\Delta s_r| & 0 \\ 0 & k_l|\Delta s_l| \end{bmatrix} = \begin{bmatrix} \sigma_{s_r}^2 & 0 \\ 0 & \sigma_{s_l}^2 \end{bmatrix}$$



Sensor noise → Position noise

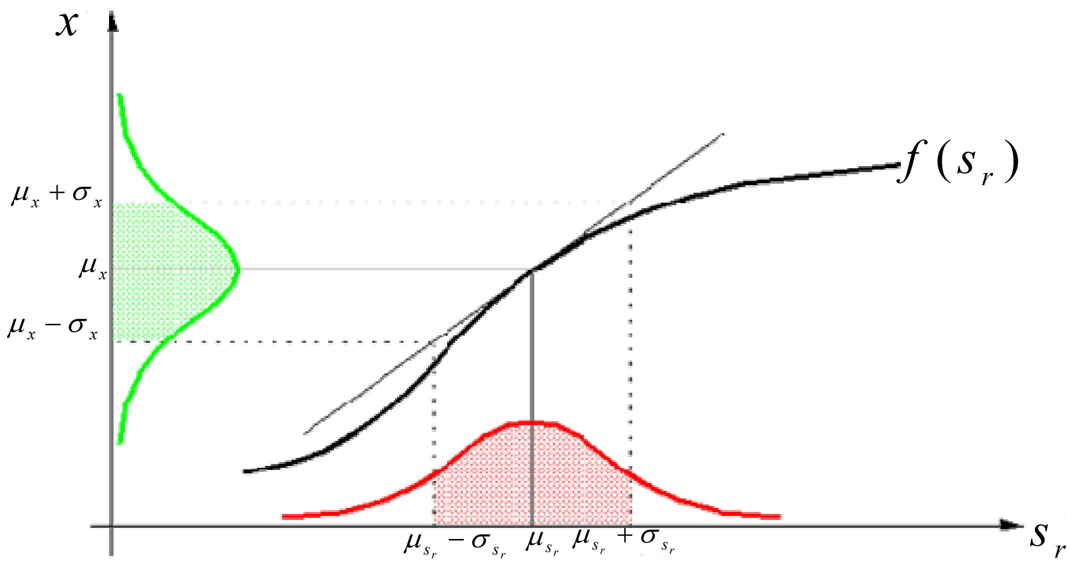
- How is the noise (2D) propagated to the position (3D)?

$$\Sigma_\Delta = \begin{bmatrix} \sigma_{s_r}^2 & 0 \\ 0 & \sigma_{s_l}^2 \end{bmatrix}$$



$$\begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 & \sigma_{x\theta}^2 \\ \sigma_{yx}^2 & \sigma_{yy}^2 & \sigma_{y\theta}^2 \\ \sigma_{\theta x}^2 & \sigma_{\theta y}^2 & \sigma_{\theta\theta}^2 \end{bmatrix} = \Sigma_p$$

- 1D to 1D example $N(\mu_{s_r}, \sigma_{s_r}) \rightarrow N(\mu_x, \sigma_x)$



- We need to linearize → Taylor Series

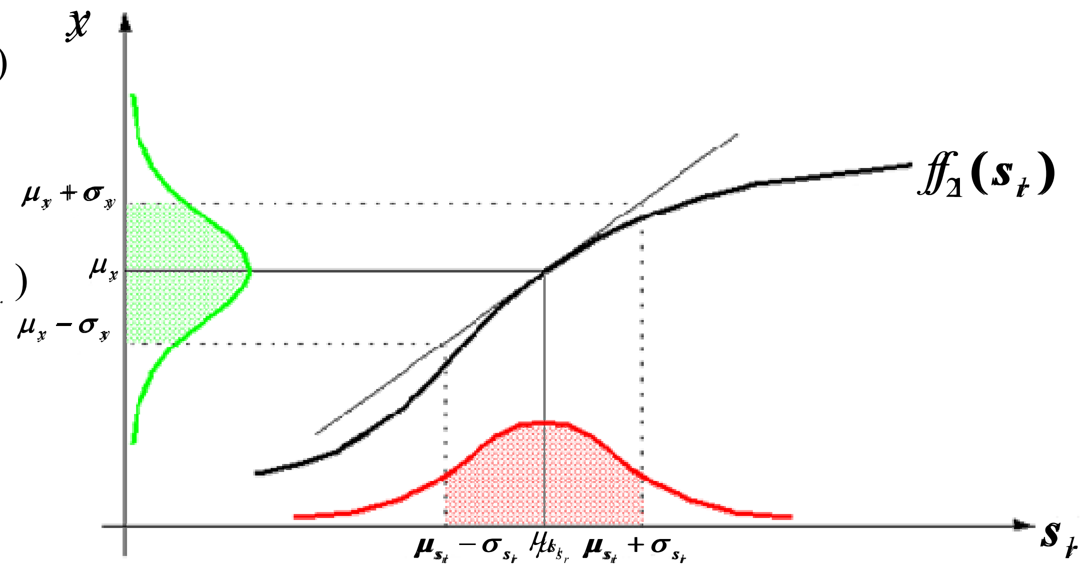
$$x \approx f(s_r)\big|_{s_r=\mu_{s_r}} \approx f(s_r) + \frac{1}{1!}\frac{\partial f}{\partial s_r}(s_r - \mu_{s_r}) + \frac{1}{2!}\frac{\partial^2 f}{\partial s_r^{\,2}}(s_r - \mu_{s_r})^2 + \ldots$$

Sensor noise → Position noise

$$x \approx f_1(s_r)\big|_{s_r=\mu_{s_r}} \approx f_1(s_r) + \frac{\partial f_1}{\partial s_r}(s_r - \mu_{s_r})$$

$$x \approx f_1(s_l)\big|_{s_l=\mu_{s_l}} \approx f_1(s_l) + \frac{\partial f_1}{\partial s_l}(s_l - \mu_{s_l})$$

$$y \approx f_2(s_r)\big|_{s_r=\mu_{s_r}} \approx f_2(s_r) + \frac{\partial f_2}{\partial s_r}(s_r - \mu_{s_r})$$



$$F_{\Delta rl} = \begin{bmatrix} \frac{\partial f_1}{\partial s_r} & \frac{\partial f_1}{\partial s_l} \\ \frac{\partial f_2}{\partial s_r} & \frac{\partial f_2}{\partial s_l} \\ \frac{\partial f_3}{\partial s_r} & \frac{\partial f_3}{\partial s_l} \end{bmatrix}$$

Jacobian

- General error propagation law

$$\Sigma_{\Delta rl} = F_{\Delta rl} \Sigma_{\Delta} F_{\Delta rl}^{T}$$

Sensor noise → Position noise

How does the state covariance Σ_p evolve over time?

- Initial covariance of vehicle at t=0:

$$\Sigma_p^{(t=0)} = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 & \sigma_{x\theta}^2 \\ \sigma_{yx}^2 & \sigma_{yy}^2 & \sigma_{y\theta}^2 \\ \sigma_{\theta x}^2 & \sigma_{\theta y}^2 & \sigma_{\theta\theta}^2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

- Additional noise at each time step Δt: $\Sigma_{\Delta rl} = F_{\Delta rl} \Sigma_{\Delta} F_{\Delta rl}^T$
- Covariance at t=1: $\Sigma_p^{(t=1)} = \Sigma_p^{(t=0)} + \Sigma_{\Delta rl} = \Sigma_{\Delta rl}$
- Covariance at t=2:

$$\Sigma_p^{(t=2)} = F_p \Sigma_p^{(t=1)} F_p^T + F_{\Delta rl} \Sigma_{\Delta} F_{\Delta rl}^T$$

$$F_p = \begin{bmatrix} \frac{\partial f_1}{\partial_x} & \frac{\partial f_1}{\partial_y} & \frac{\partial f_1}{\partial_\theta} \\ \frac{\partial f_2}{\partial_x} & \frac{\partial f_2}{\partial_y} & \frac{\partial f_2}{\partial_\theta} \\ \frac{\partial f_3}{\partial_x} & \frac{\partial f_3}{\partial_y} & \frac{\partial f_3}{\partial_\theta} \end{bmatrix}$$

Sensor noise → Position noise

Recipe

Precompute:

- Determine sensor noise $\Sigma_{\Delta rl}$
- Compute mapping sensor noise to system noise $F_{\Delta rl}$
- Compute mapping system noise to system noise F_p

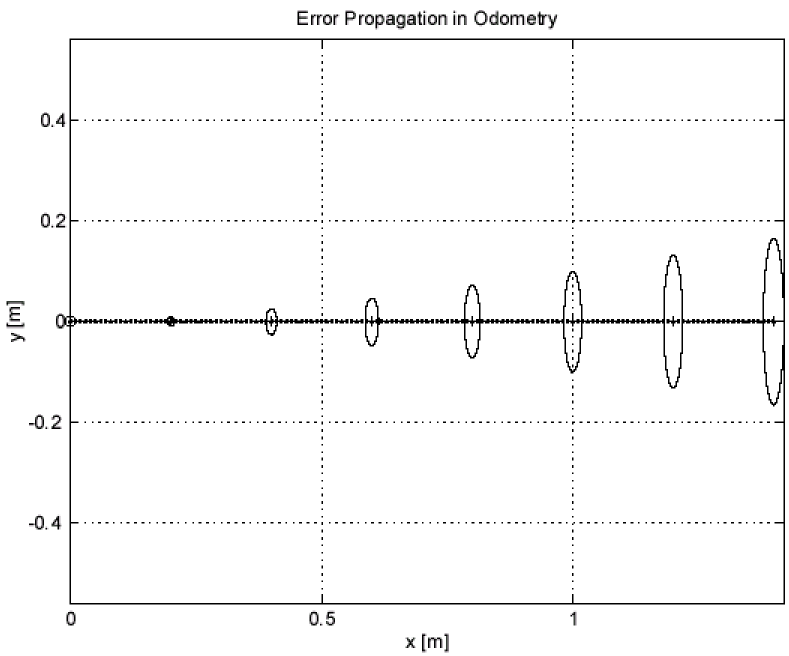
Initialize:

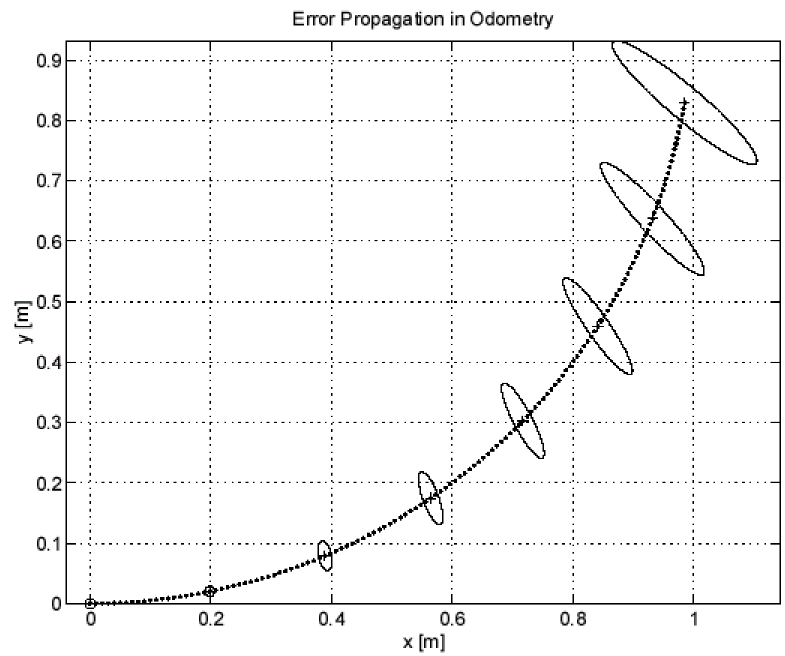
- Initialize $\Sigma_p^{(t=0)} = [0]$

Iterate:

$$\Sigma_p^{(t=2)} = F_p \Sigma_p^{(t=1)} F_p^T + F_{\Delta rl} \Sigma_{\Delta rl} F_{\Delta rl}^T$$

Classical 2D representation





Courtesy of R. Siegwart and R. Nourbakhsh

Conclusion

Take Home Messages

- Perception-to-action loop is key in robotics, several sensor and actuator modalities
- Experimental work can be carried out with real and realistically simulated robots
- A given behavior can be obtained with different control architectures
- There are several localization techniques for indoor and outdoor systems
- Each of the localization methods/positioning system has advantage and drawbacks.
- Odometry allows for computing the absolute position of a robot using only on-board, cheap sensors; however, its accuracy decreases with time (cumulative error) if not reset

Additional Literature – Week 3

Books

- Braitenberg V., “Vehicles: Experiments in Synthetic Psychology”, MIT Press, 1986.
- Siegwart R. and Nourbakhsh I. R., “Introduction to Autonomous Mobile Robots”, MIT Press, 2004.
- Arkin R. C., “Behavior-Based Robotics”. MIT Press, 1998.
- Everett, H. R., “Sensors for Mobile Robots, Theory and Application”, A. K. Peters, Ltd., 1995
- Choset H., Lynch K. M., Hutchinson S., Kantor G., Burgard W., Kavraki L., and Thrun S., “Principles of Robot Motion”. MIT Press, 2005
- Thrun S., Burgard W., and Fox D., “Probabilistic Robotics”, MIT Press, 2005.
- Arnaud P., “Des moutons et des robots”, Presses Polytechniques et Universitaires Romandes, 2000.
- Maybeck P. S. “Stochastic Models, Estimation, and Control”, Academic press, 1979, ch. 1. Out of print but available at: *cs.unc.edu/~welch/media/pdf/**maybeck**_ch1.pdf*

PhD Theses

- Asadpour M., “Behavior Design in Microrobots: Hierarchical Reinforcement Learning under Resource Constraints”, EPFL PhD Thesis Nr. 3682, Lausanne, Switzerland, November 2006.

Lecture notes

- Merminod B., “Méthodes quantitatives I” (in French), Environmental and civil engineering curriculum, BS 5.