# EE Special Course
# Design of Intelligent SoC Robot


Hoi-Jun Yoo

**KAIST**



Introduction

# Outline

*-Design of Intelligent SoC Robot-*

1. *Introduction*
2. *Software Algorithms for Robot*
3. *Embedded System*
4. *Course Preview*
5. *Intelligence Robot War*

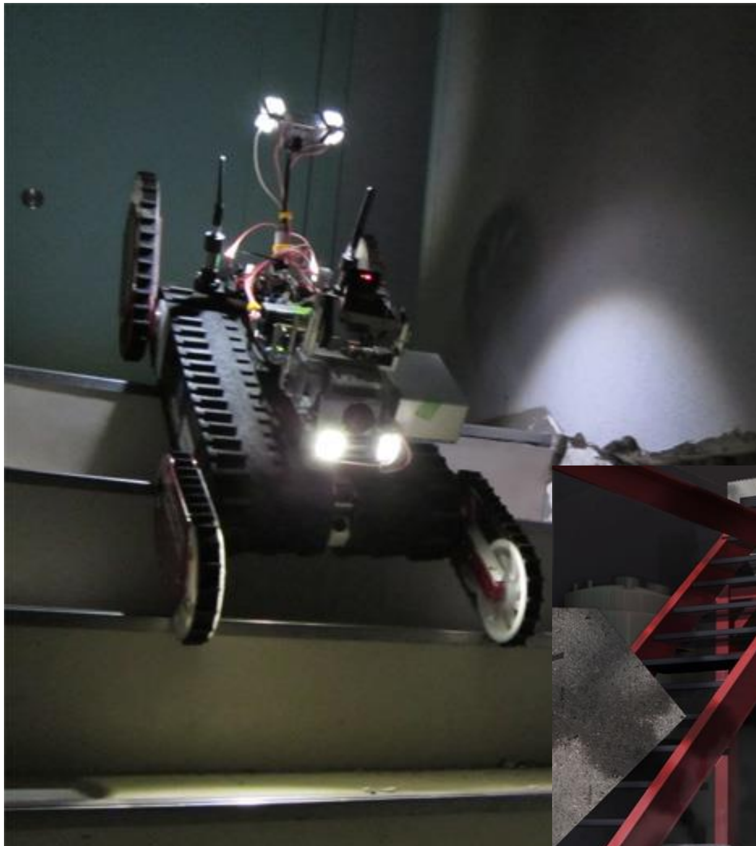
# Outline

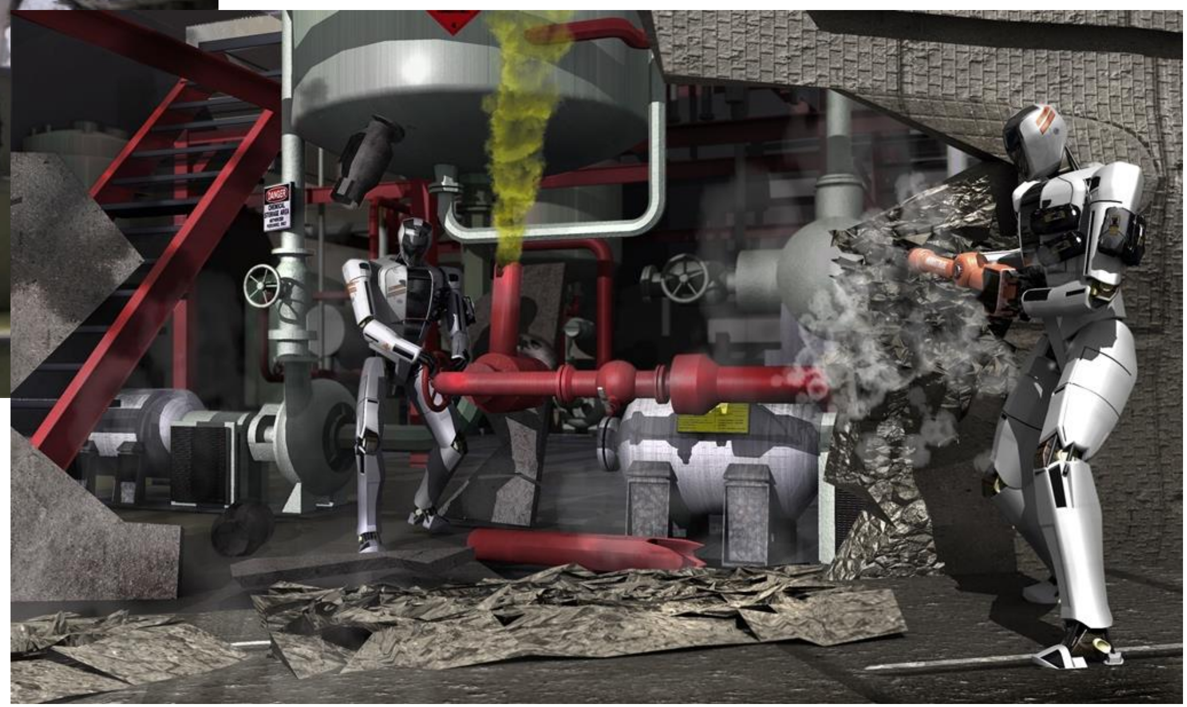
*-Design of Intelligent SoC Robot-*

1. *Introduction*
   - *What will be a future robot?*
2. *Software Algorithms for Robot*
3. *Embedded System*
4. *Course Preview*
5. *Intelligence Robot War*

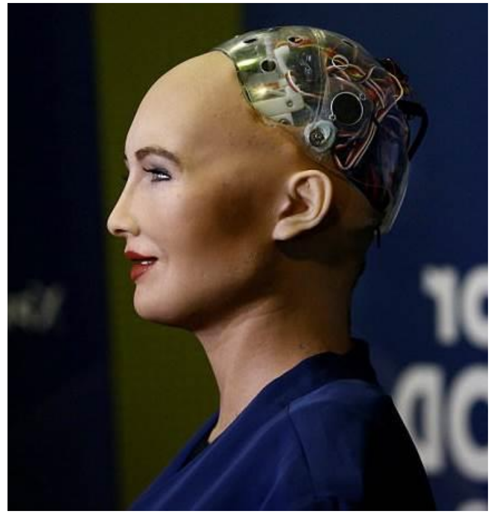
# Robot in Industry

# Robot for the Dangerous Place/Space

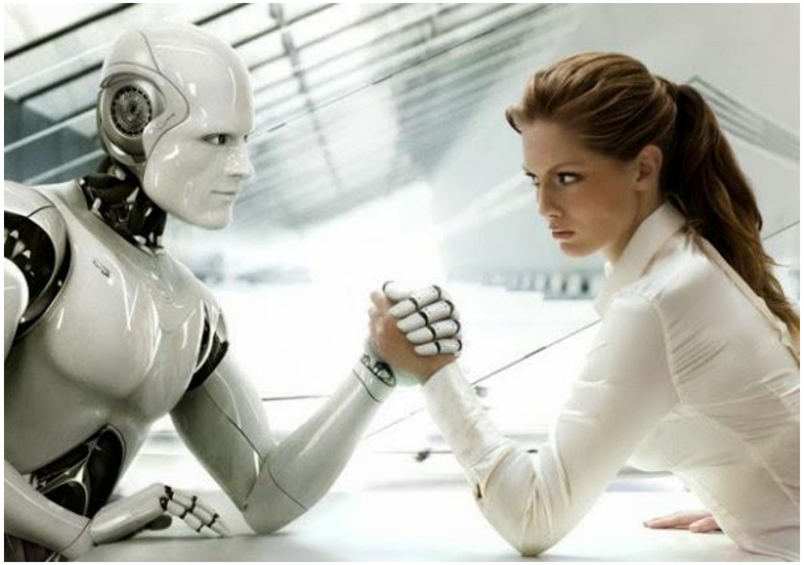


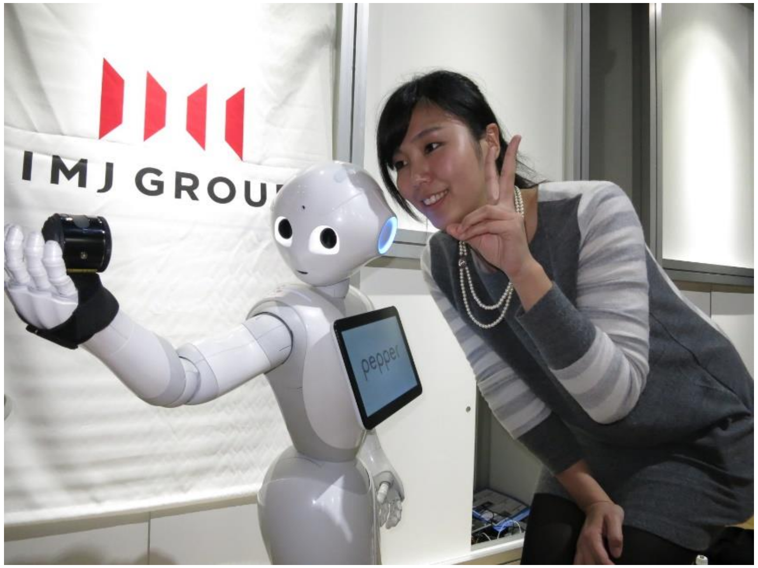


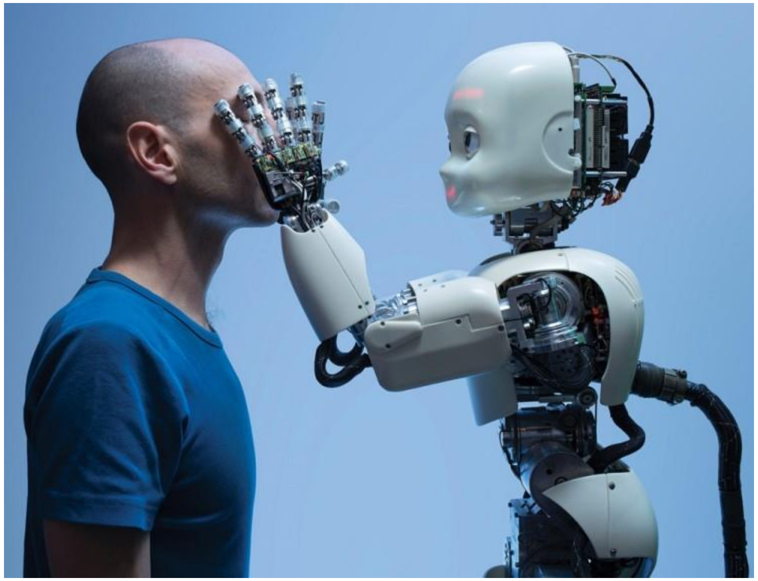
# Humanoid Robot

# Human-Robot Collaboration





# Types of Robots

1. Manipulators

   Anchored to the workplace.

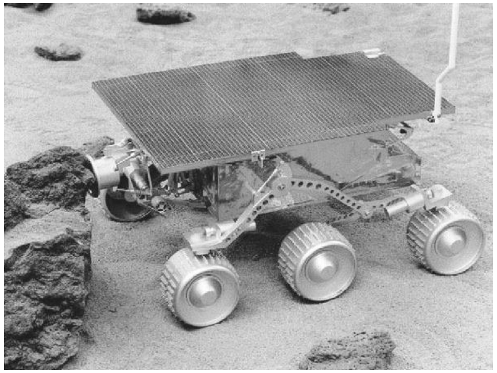
   Common industrial robots.

2. Mobile Robots

   Move using wheels, legs, flies, swims, etc.

   Examples: delivering food in hospitals,

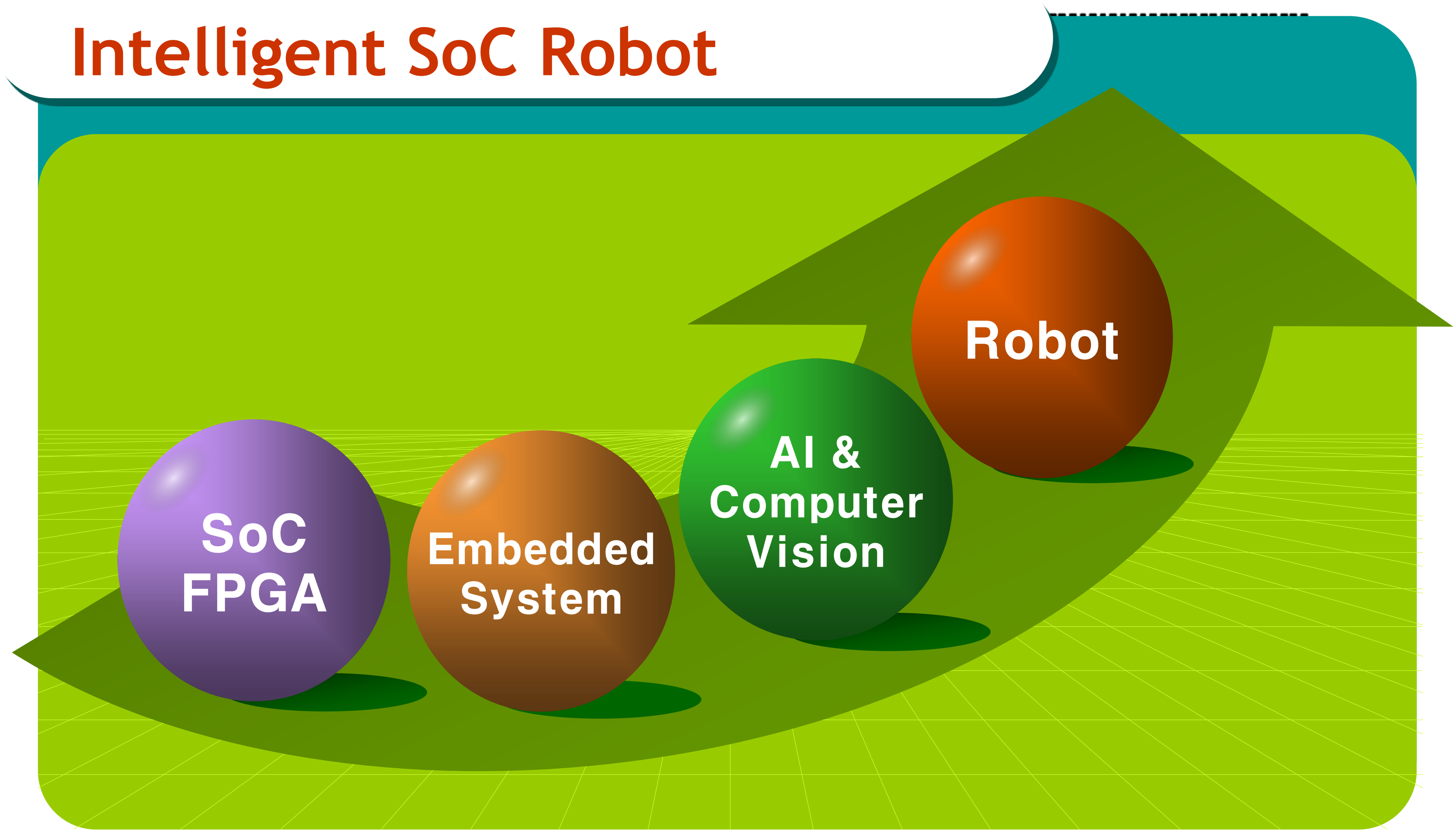
   autonomous navigation, surveillance, etc.



# Types of Robots

3. Hybrid (mobile with manipulators)
   Examples: humanoid robot
   (physical design mimics human torso)
   Recently very popular.


# Autonomous Robot HW & SW



# Outline

## *-Design of Intelligent SoC Robot-*

1. *Introduction*

2. *Software Algorithms for Robot*
   - *What is required for mobile SoC robot?*

3. *Embedded System*

4. *Course Preview*

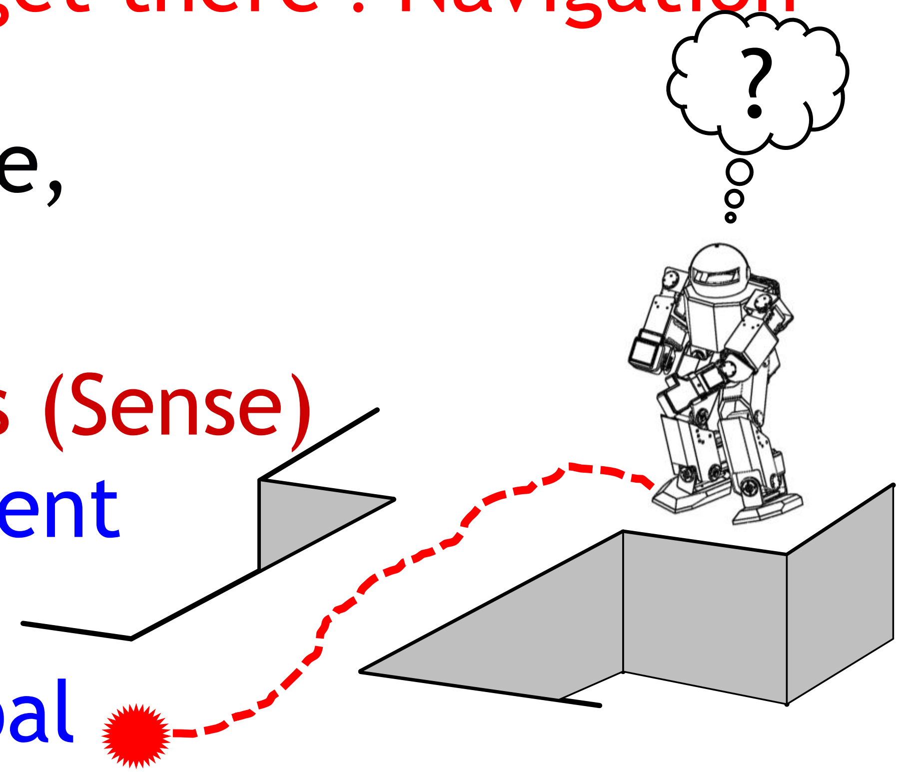
5. *Intelligence Robot War*

# 3 Fundamental Questions

1. Where am I : Current Position
2. Where am I going : Destination
3. When and How do I get there : Navigation

In order to answer these,
the robot must

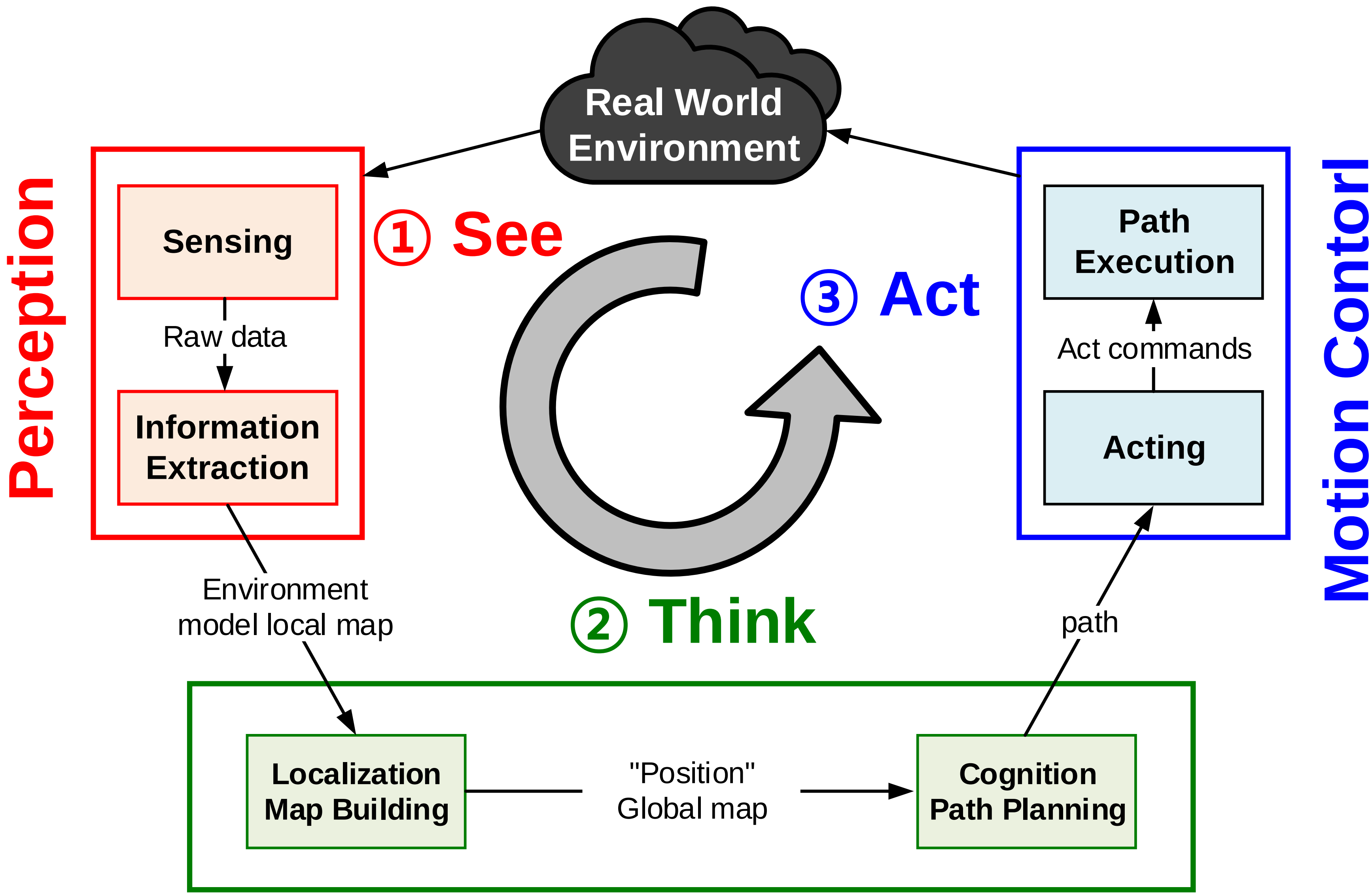
1. Make measurements (Sense)
2. Model the environment
3. Localize itself
4. Plan a path to its goal
5. Navigate itself to its goal



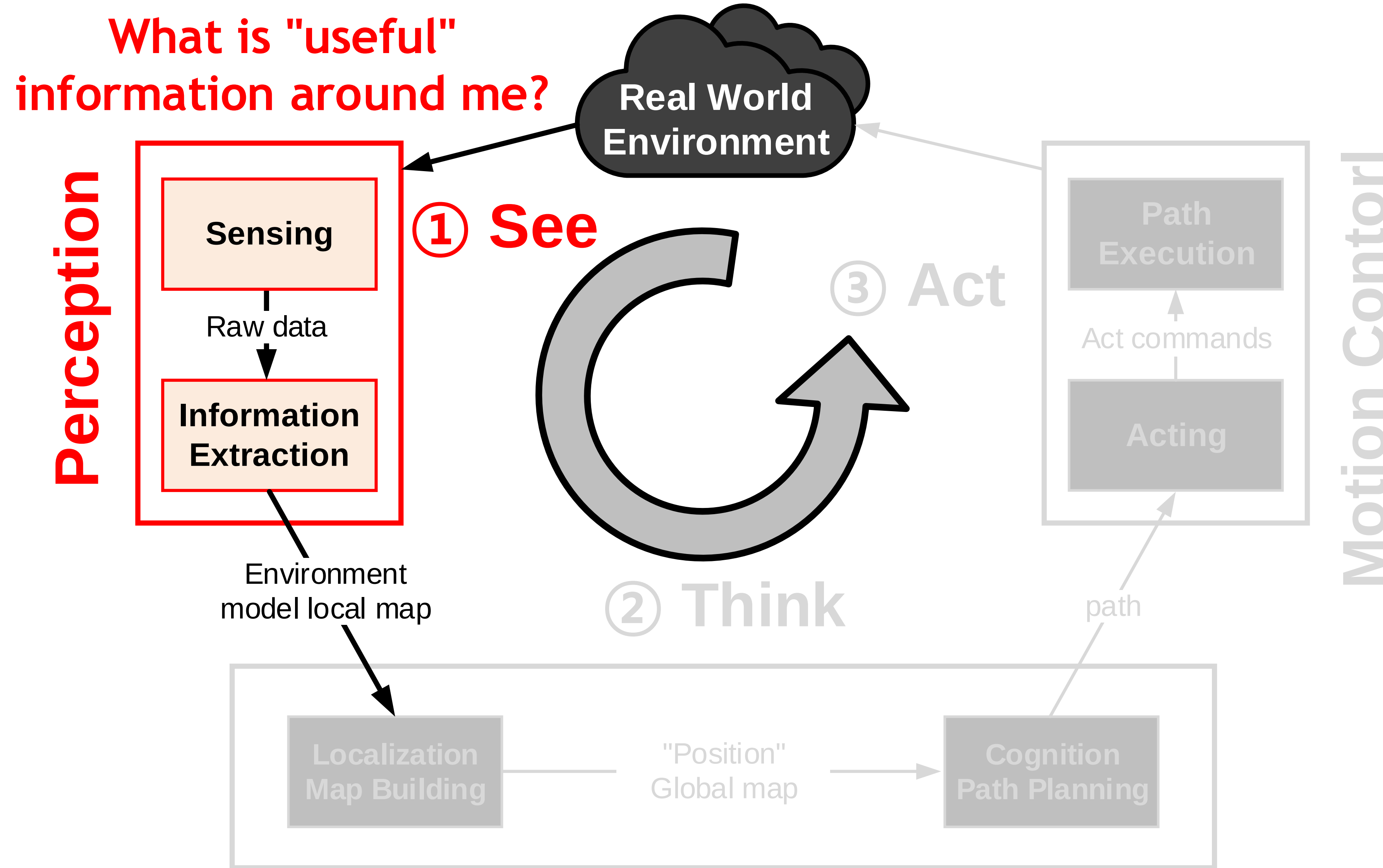
# Robot SW Language Level

| | | |
|---|---|---|
| 7 | Robot Scenario Vocabulary | Robot Role |
| 6 | Robot Task Vocabulary | Robot Role |
| 5 | Robot Situation Language | Robot Role |
| 4 | Robot Base Language | Robot Capabilities |
| 3 | Coding Languages (e.g., C, C++, Java, Python) & Visual Language (e.g., LabView) | Robot Capabilities |
| 2 | μ-Controller Assembly Language | HW Instructions Capabilities |
| 1 | μ-Controller Binary, Hex, Octal Language | HW Instructions Capabilities |

# The See-Think-Act (STA) cycle



# The first stage : See

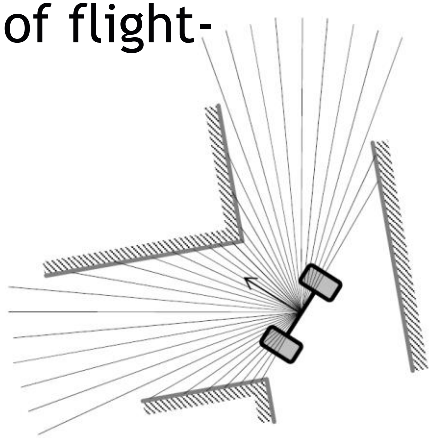


# Perception : Sensing

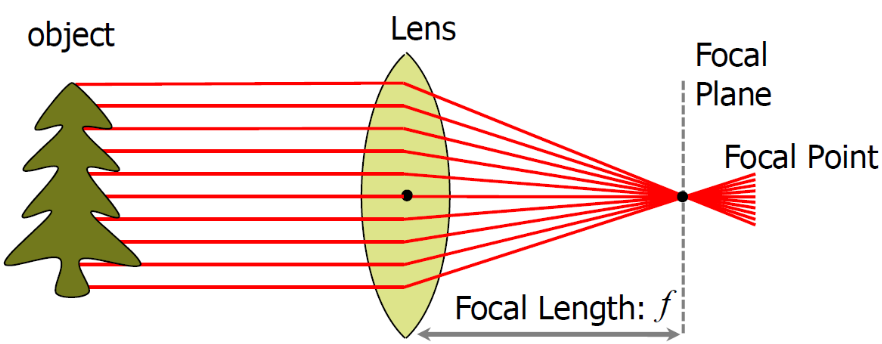
Get all information from environment

## Laser scanner

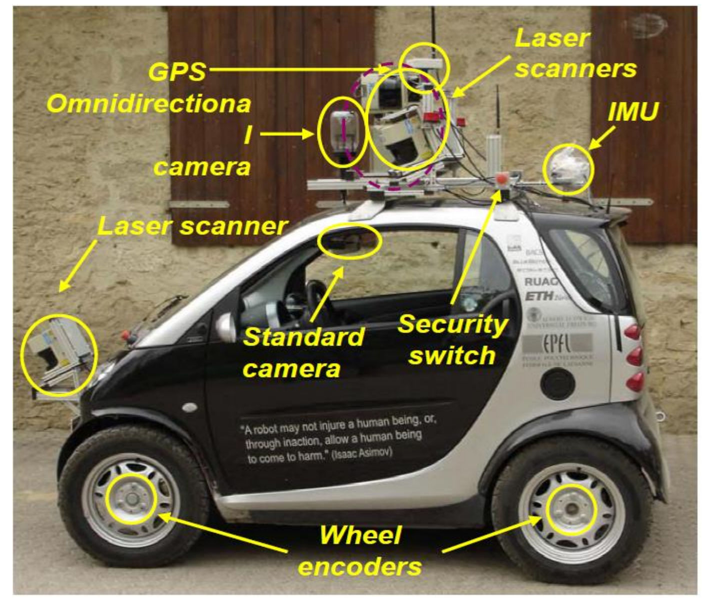
-Time of flight-

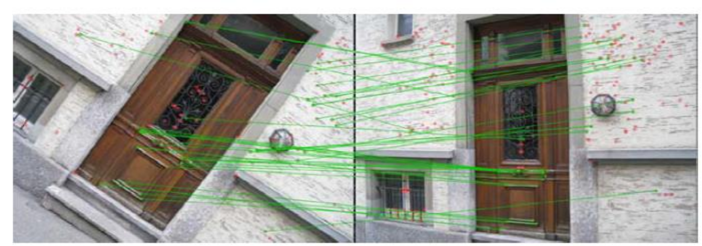


## Camera



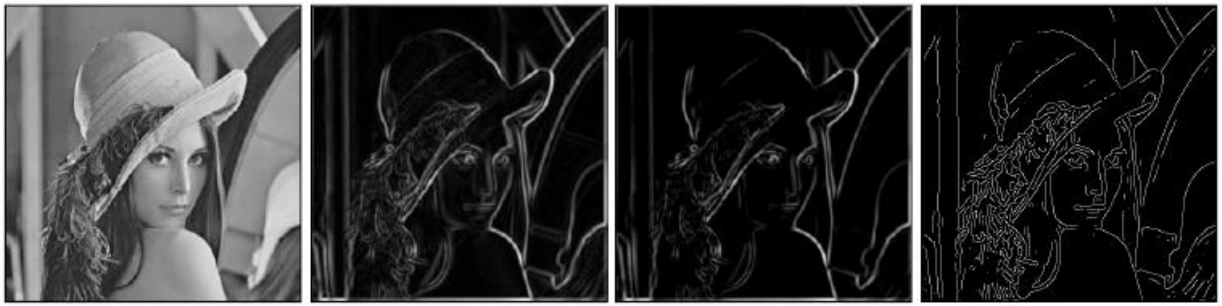

## Example of multi-sensor autonomous car



# Perception : information extraction

Extract "key" information for "think" stage



- Filtering
- Edge Detection

- Keypoint Features
  - Features that are invariant to rotation, scaling, viewpoint, illuminations

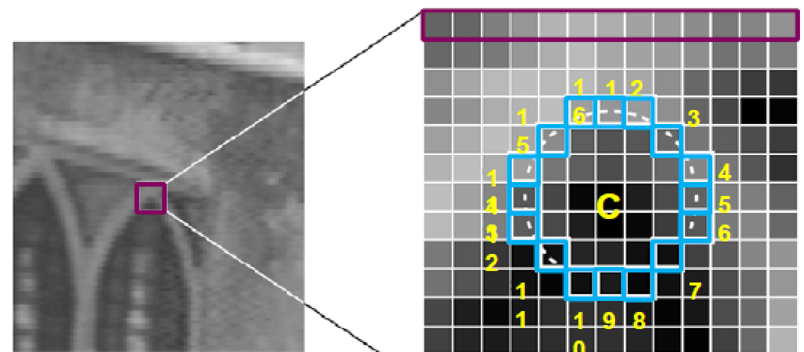
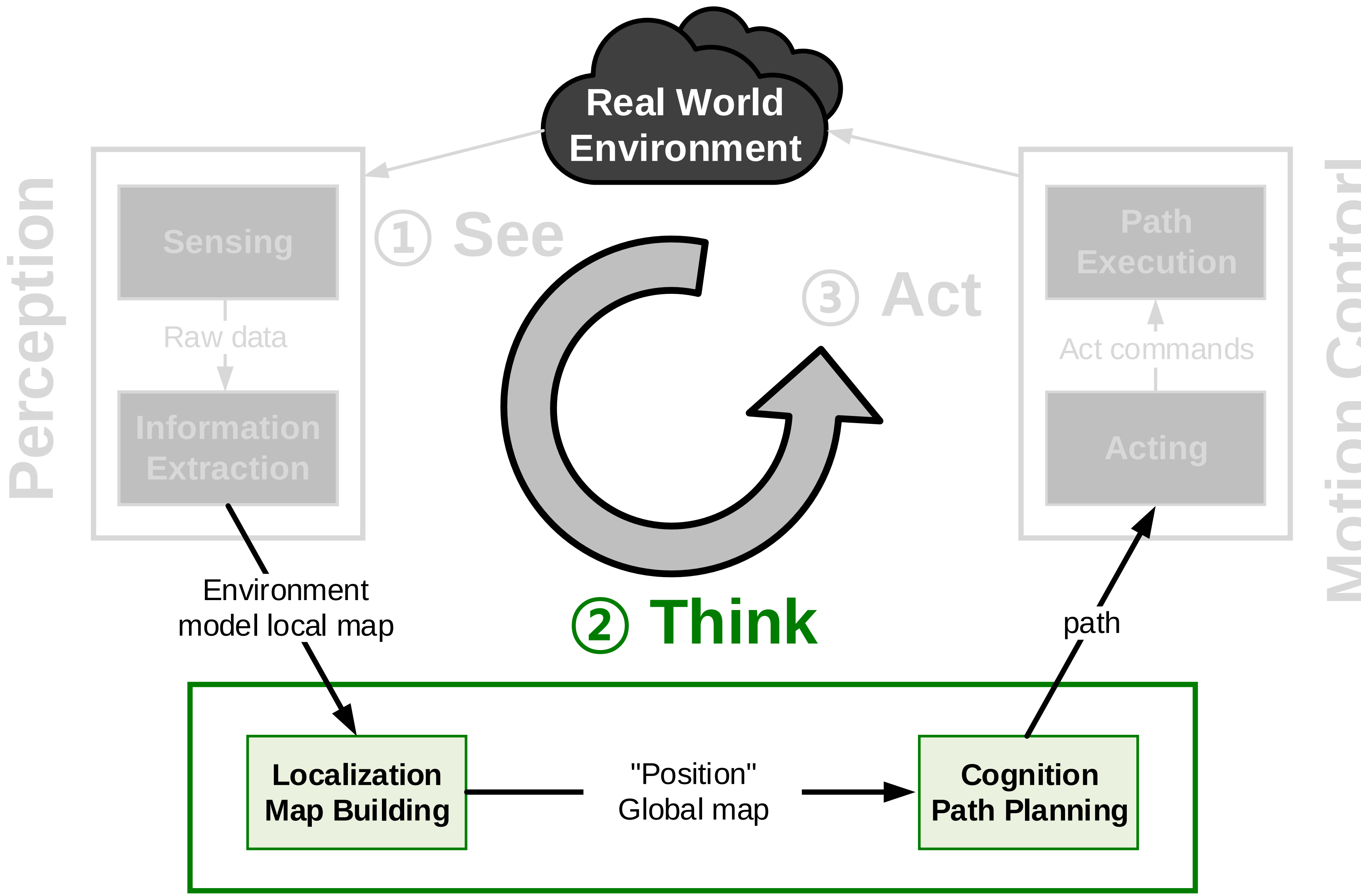



Image from [Rosten et al., PAMI 2010]


- Keypoint matching
  - Object detection, etc...

# The second stage : Think



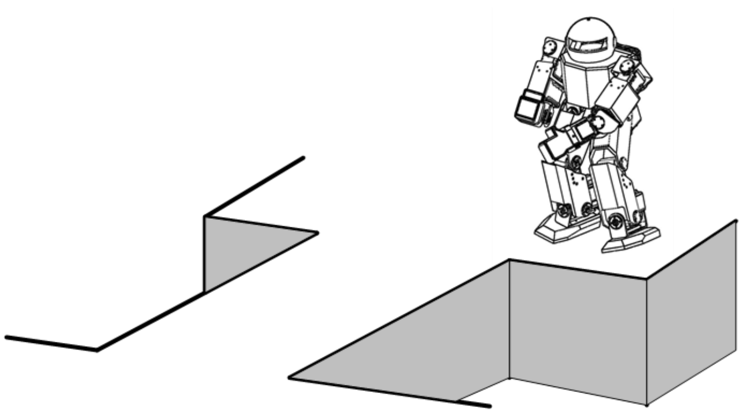

Where am I?

How can I go there?

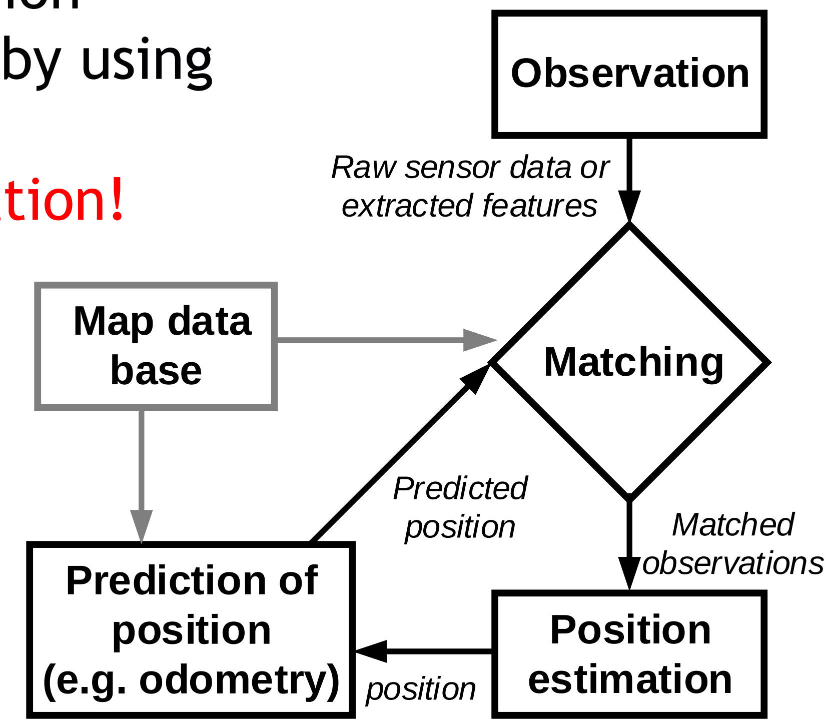
# Think : Localization

Before thinking about how to go to destination, Autonomous robot should know “where am I?”

- Flow of localization
  - 1. Estimate current position
  - 2. Predict next position (by using odometry, etc…)
  - 3. Matching with observation!
  - 4. Back to 1.



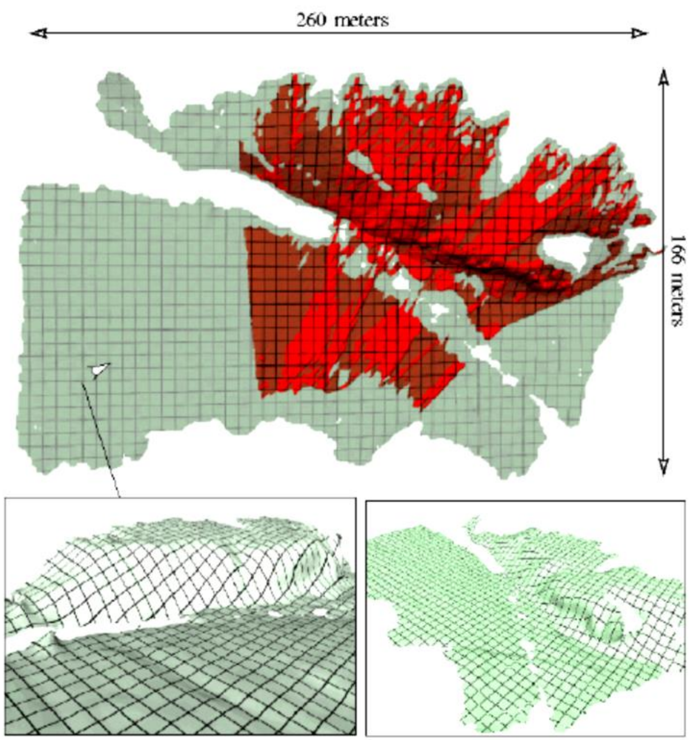
<Where am I ?>

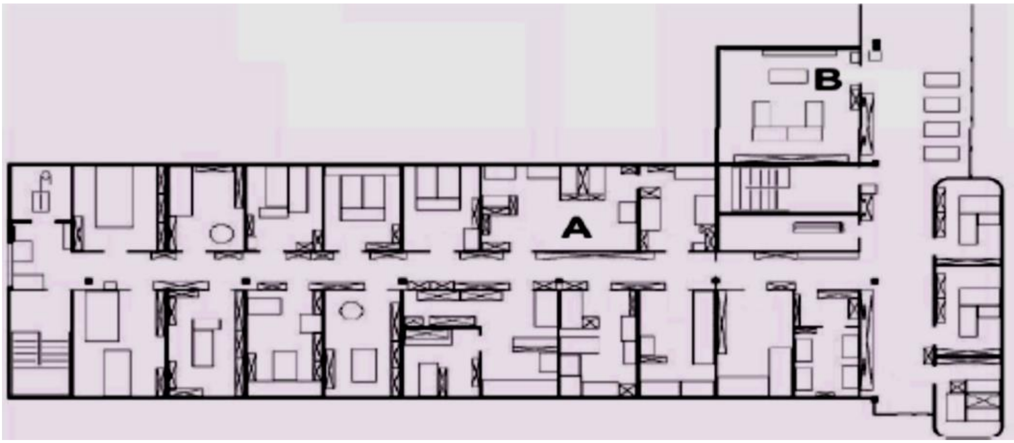



<General flow of localization>

# Think : Mapping

- Maps are required to help a robot get from point A to B
- Map representations can be continuous or discrete
- Maps can be built a priori and/or dynamically



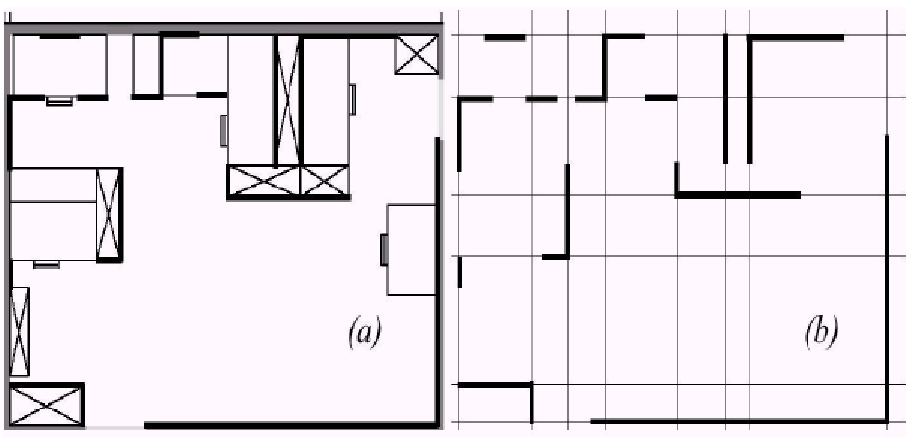
# Think : Mapping (cont.)

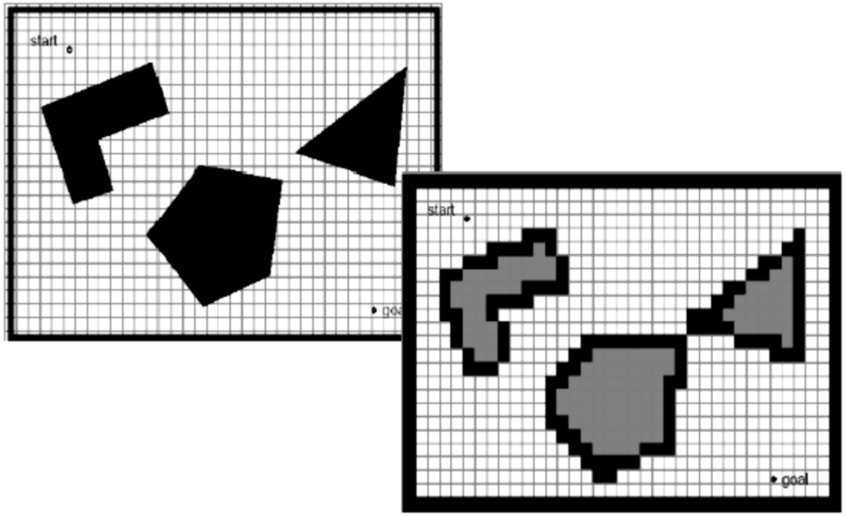
Continuous representations

- Maps made from line segments
- Matching requires good line segmentation
  - Not useful in cluttered environments
- Computationally expensive
- Good data compression
- Not useful outdoors

Discrete representations

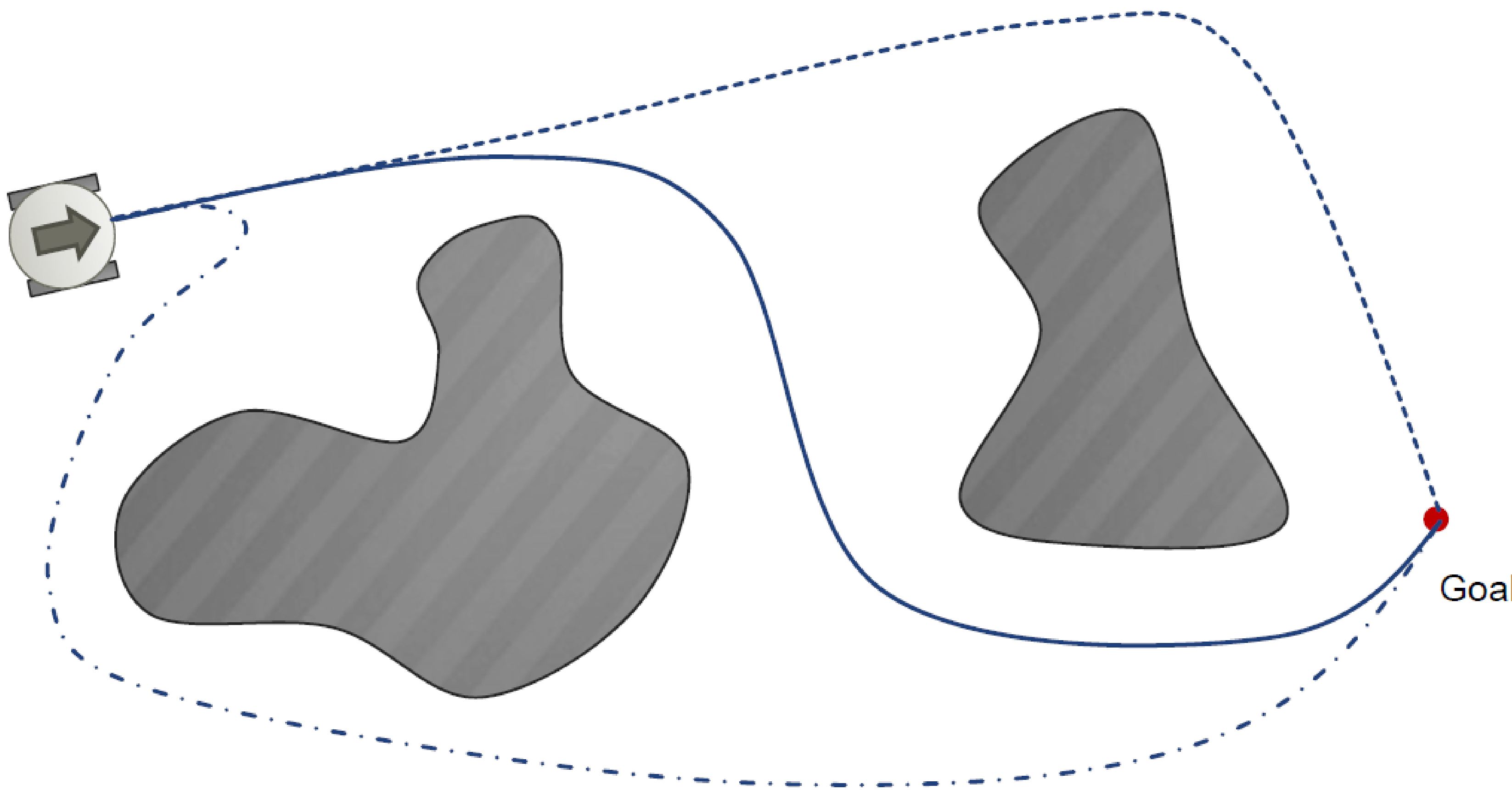
- Either fixed cell or adaptive cell size
- Suffers from aliasing, insufficient resolution
  - Can narrow passages
- Computationally more efficient
- Usually large memory footprint



# Think : Cognition

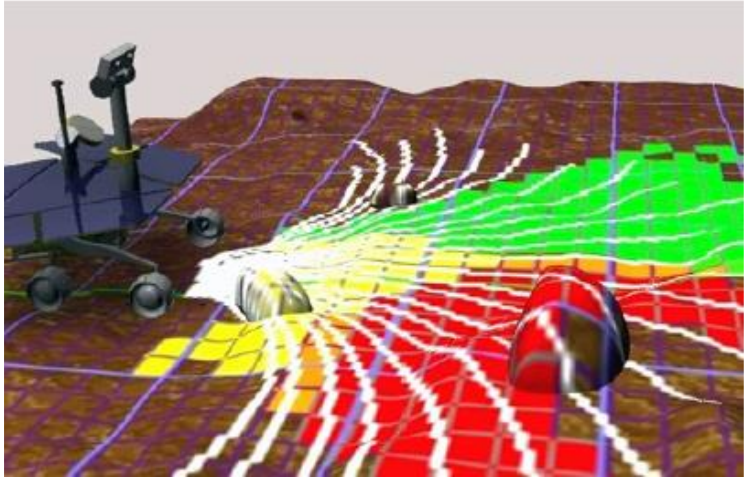
Where am I going? How do I get there?



# Think : Path Planning

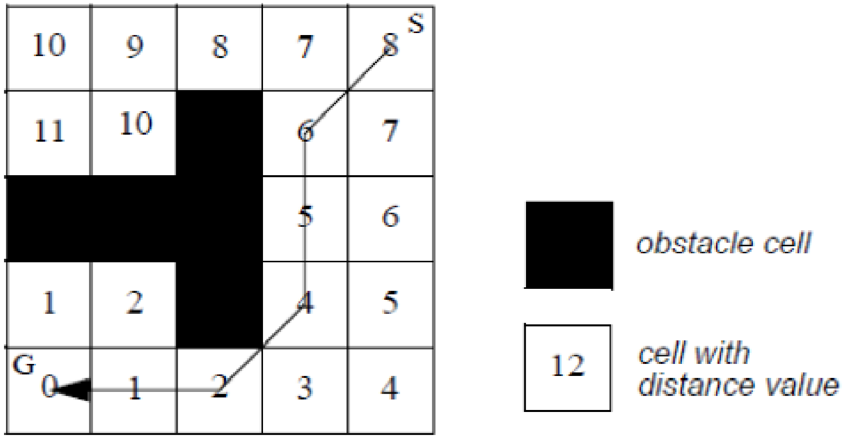
Given a map, now get from point A to B

- A Priori maps a good start
- Local vs Global
- Local path planning with obstacle detection and avoidance helps us to do it safely
- Global path planning helps us get from A to B

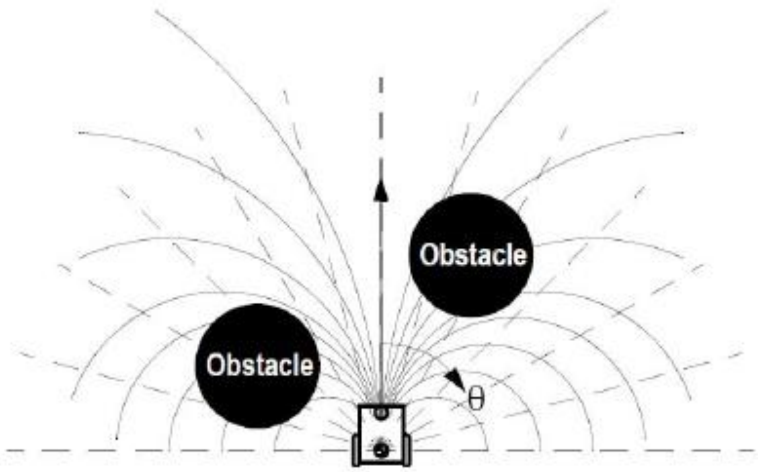


# Think : Path Planning (cont.)

- Global path planning
  - Graph search
  - “How can I go to final destination?”




- Local path planning
  - Local collision avoidance
  - “How can I avoid objects in front of me?”



# Think : Navigation

- Keeping a robot on track is not easy.
- Use a controller to keep the robot on track.
- Controllers that provide a force in negative proportion to the observed error are known as P controllers.

# Think : Navigation

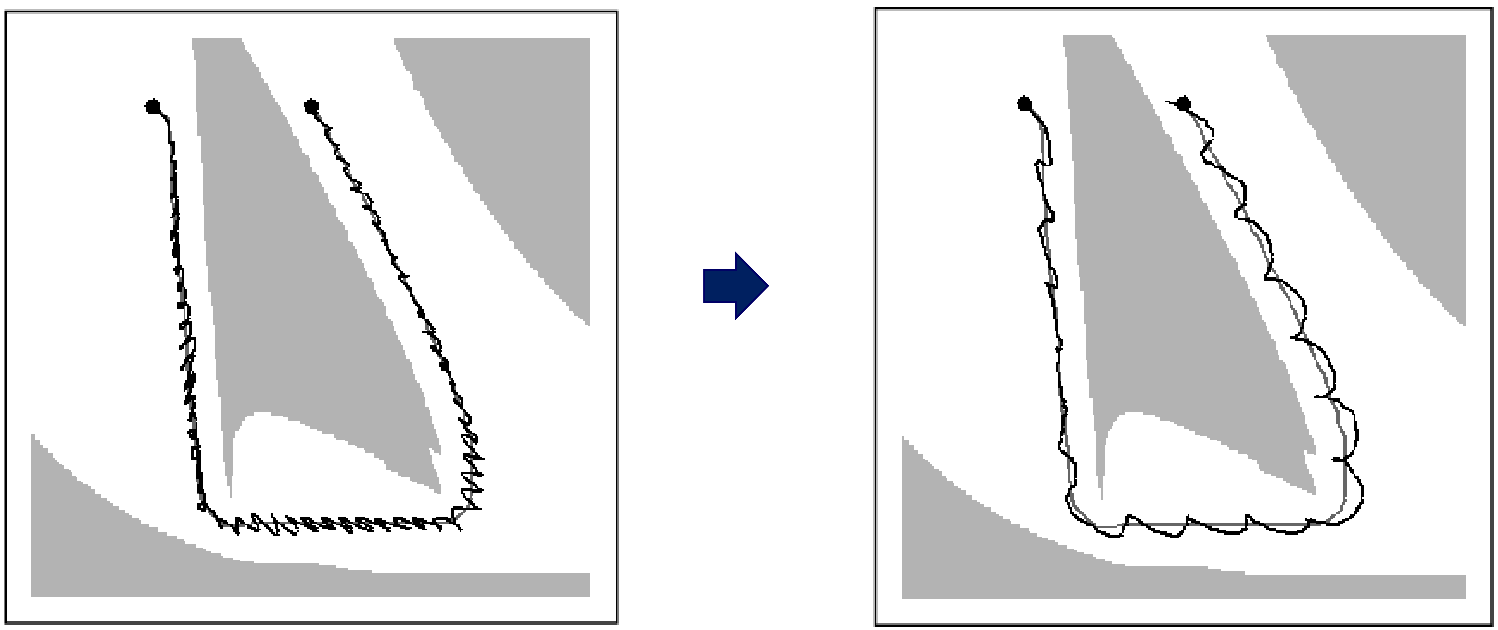
Obstacle detection

- Detect obstacles in our sensor or map data and place in the map
- Sensor

Let y(t) be the reference path.
The control generated by the controller has the form:

$$a(t) = K\,(\,y(t) - x(t)\,)$$

(K: gain parameter)



# Think : Navigation

**Obstacle detection**

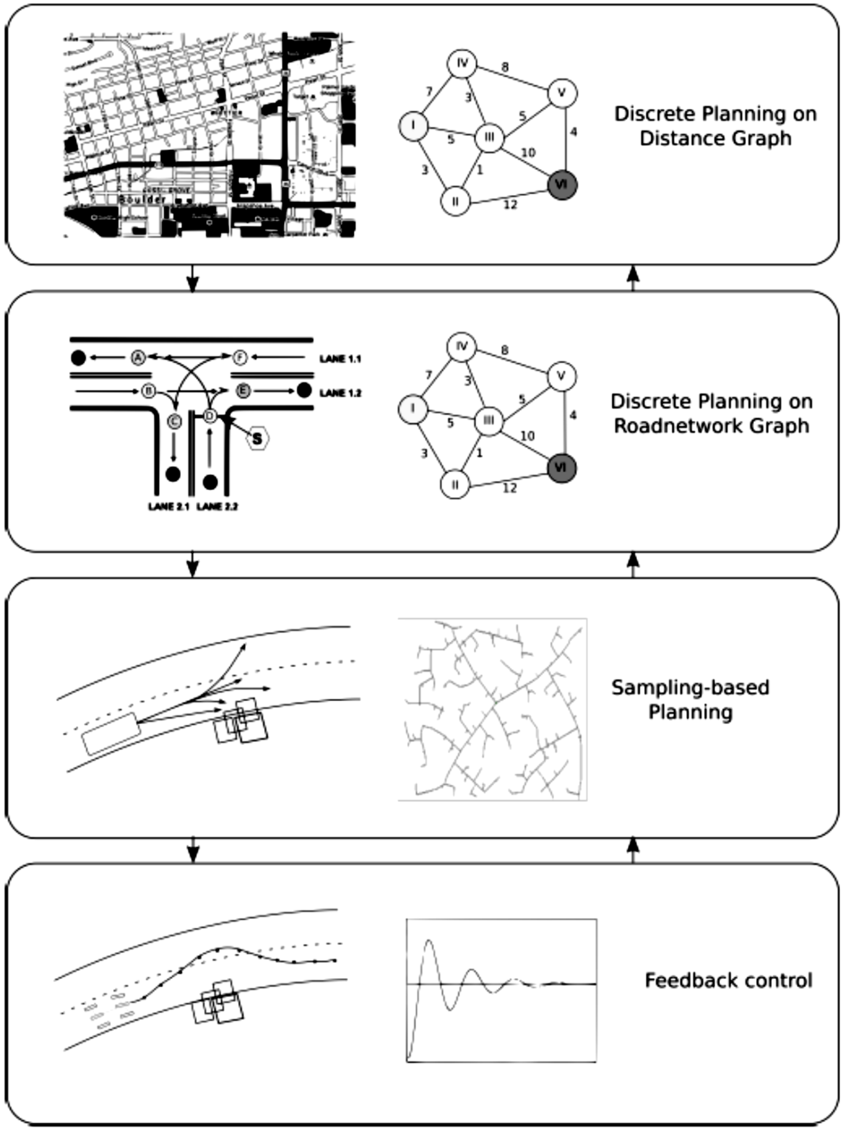
- Detect obstacles in our sensor or map data and place in the map
- Sensor measurements are analyzed for hazardous regions
  - Hazards can include barriers, slope, roughness, etc.

**Obstacle avoidance**

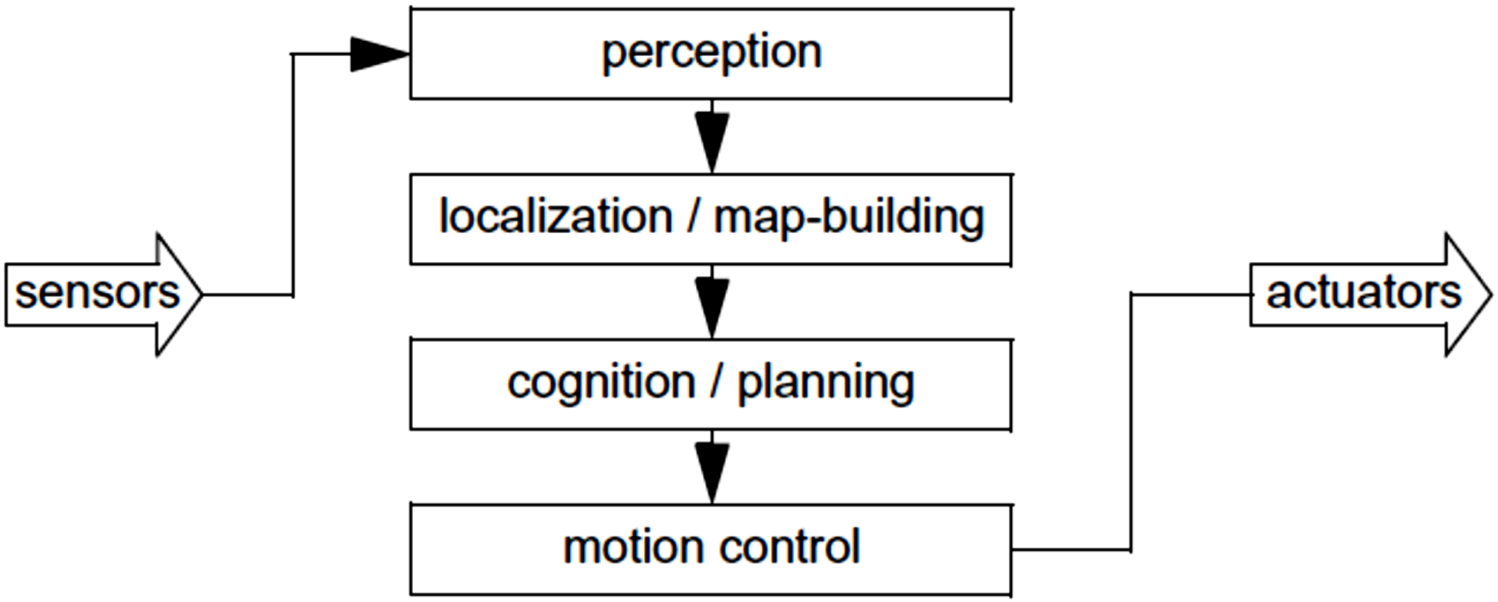
- Path planning is required to avoid hitting obstacles in the map
- Similar to global path planning except more dynamic

# Think : Navigation

# Path planning hierarchy



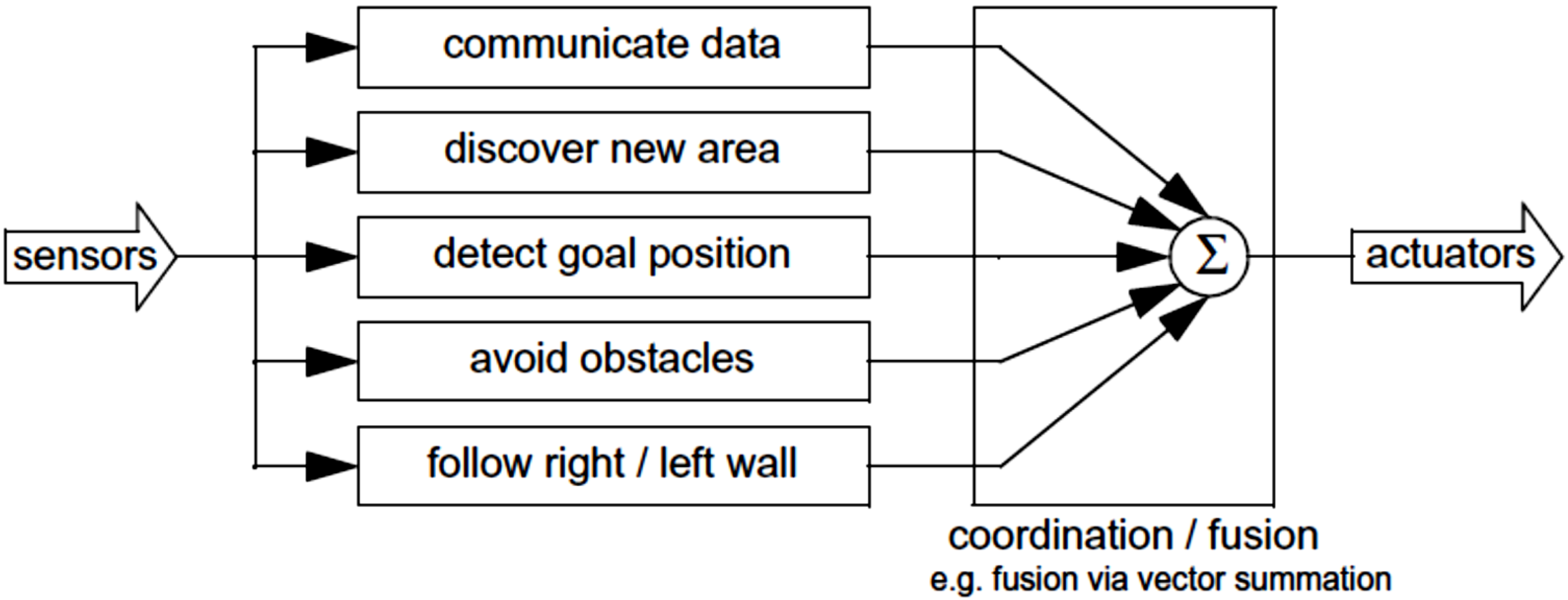
# Map-based navigation



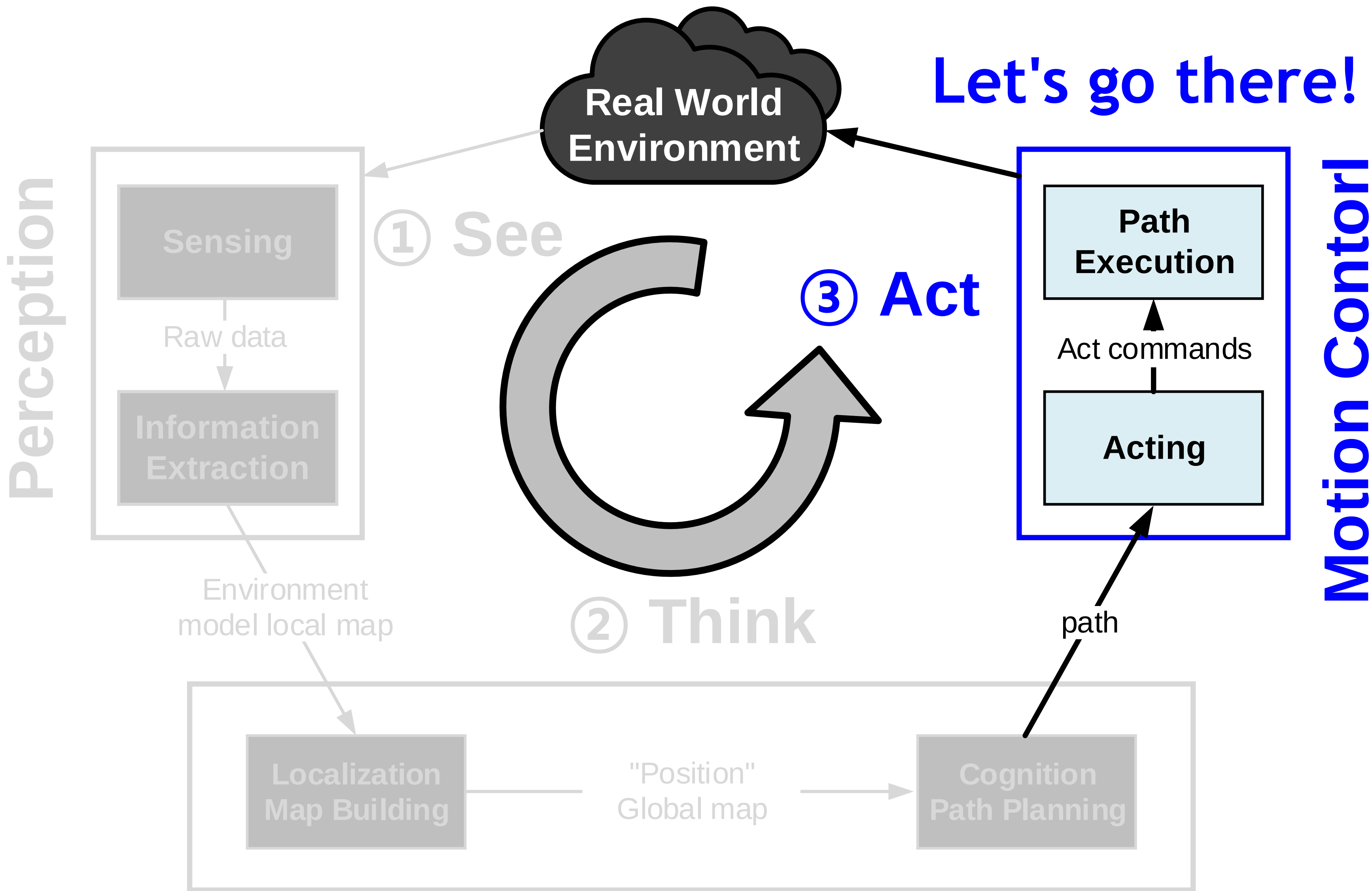
# Behavior-based

- Sensors and Effectors are noisy and information limited.
- Map is useless: No localization and path planning
- There exists a procedural solution to the particular navigation problem at hand.
  - Quick implementation for a single environment with a small number of goal positions
  - Location specific. Robot hardware specific. Fine-tuning required

# Behavior-based navigation



# The final stage : Act

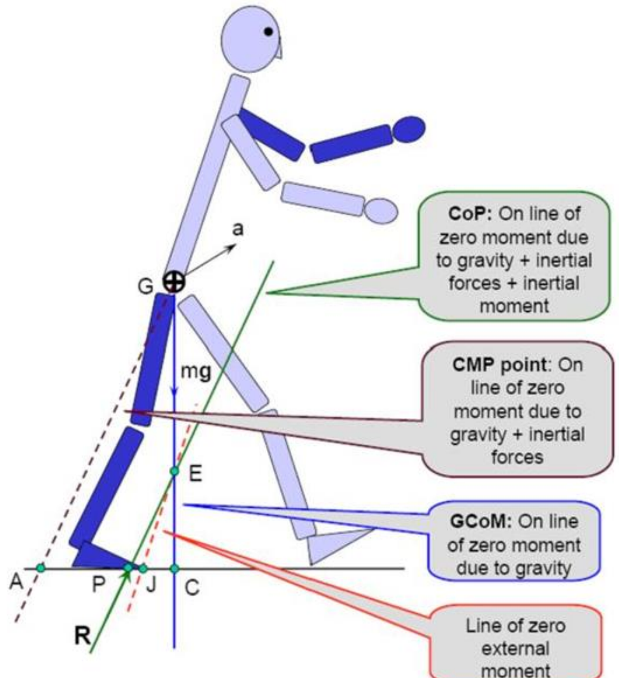


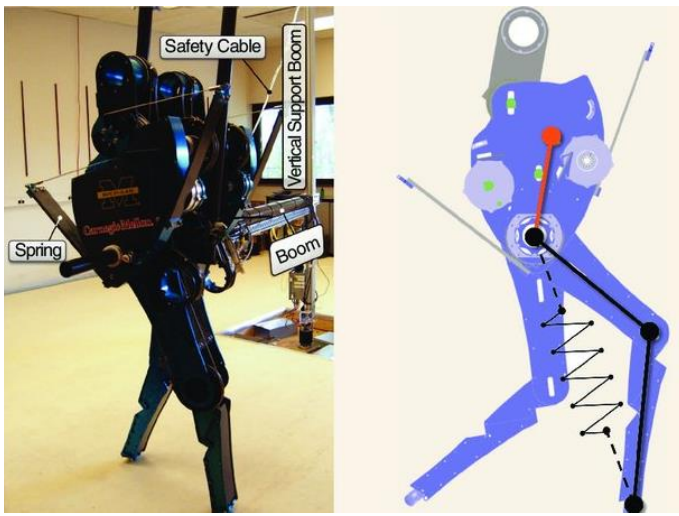
# Act : Motion Control

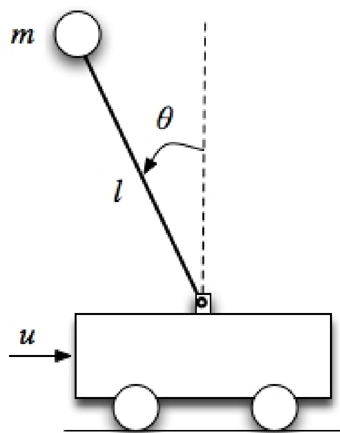
Stable Gait: contact is realized only with the soles of the foot or feet

1. Center of Mass (CoM)
2. Floor Projection of the CoM (GCoM, FCoM)
3. Zero-Moment Point (ZMP, CMP)
4. Center of Pressure (CoP)
5. ZMP only coincides with CoP

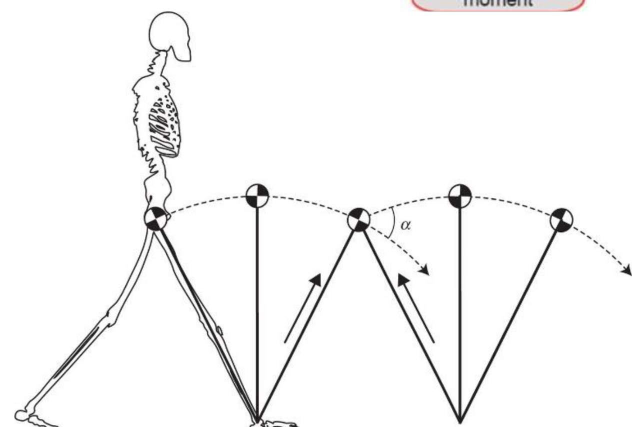
if the biped is dynamically stable




Inverted Pendulum




Spring loaded IP

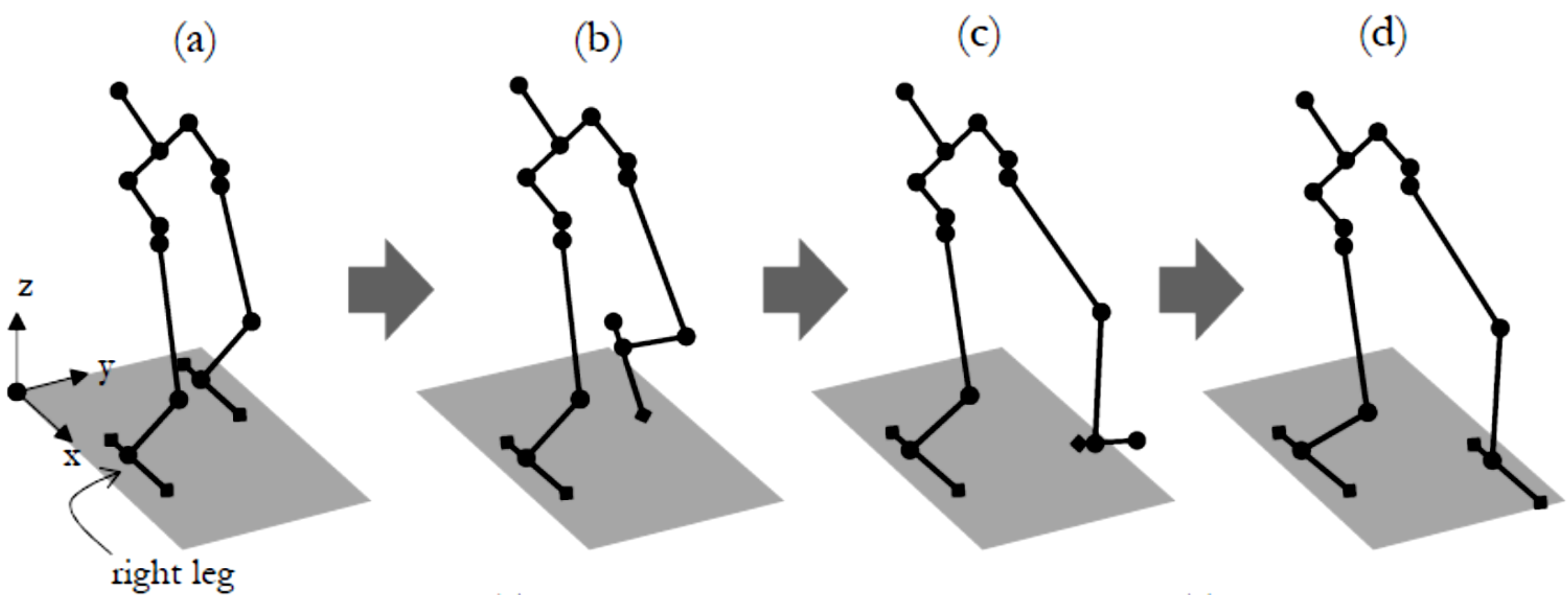
# Act : Motion Control

- Initial posture → intermediate posture → final posture

Obtain all joint motions

1. Determine Stable Postures
2. Design trajectories from one posture to another

(Walking Primitives and Inverse Differential Kinematics)



# Outline

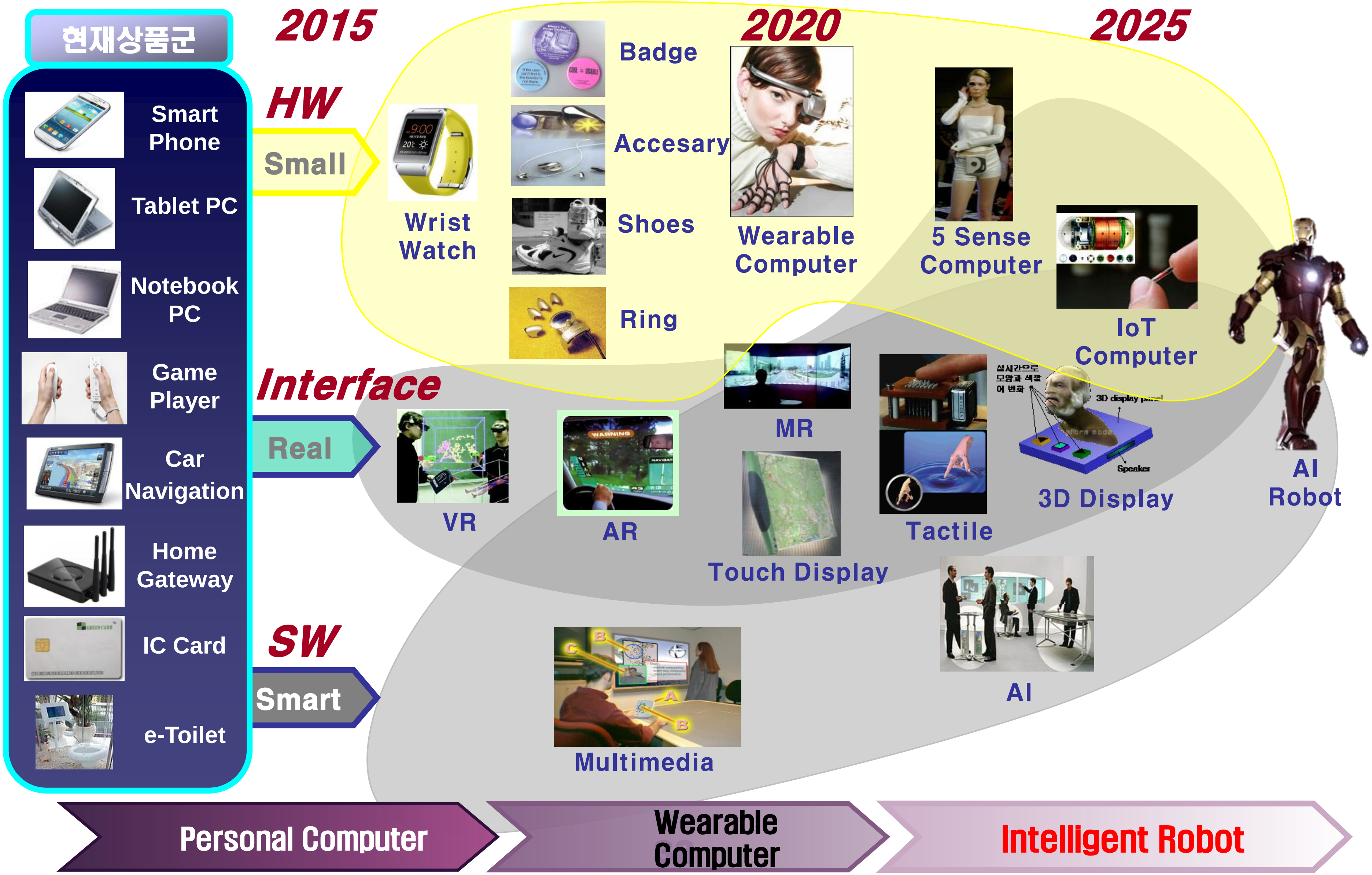
*-Design of Intelligent SoC Robot-*

1. *Introduction*
2. *Software Algorithms for Robot*
3. *Embedded System*
    - *Trend of Hardware Technology*
4. *Course Preview*
5. *Intelligence Robot War*

A Lot of algorithms required for robot…

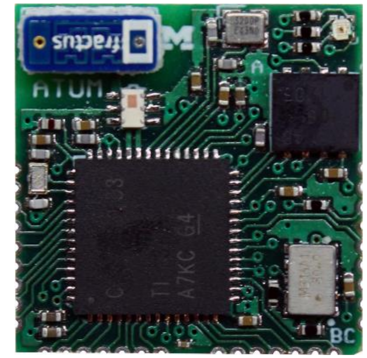
**➔ Hardware (embedded system) required for algorithms!!**

# Computer Evolution Trends

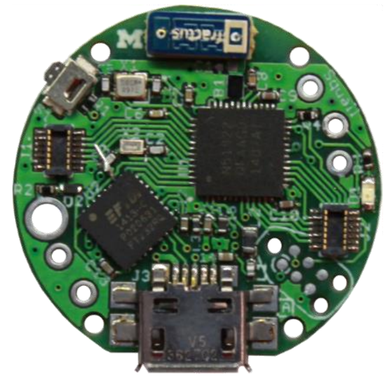


# Established comms interfaces: 802.15.4, BLE, NFC

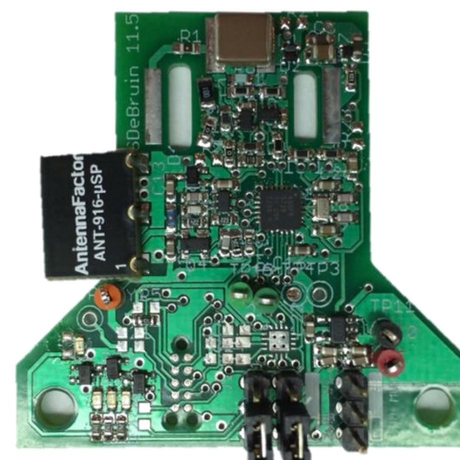
- IEEE 802.15.4 (a.k.a. "ZigBee" stack)
  - Workhorse radio technology for sensornets
  - Widely adopted for low-power mesh protocols
  - Middle (6LoWPAN, RPL) and upper (CoAP layers)
  - Can last for years on a pair of AA batteries



- Bluetooth Low-Energy (BLE)
  - Short-range RF technology
  - On phones and peripherals
  - Can beacon for years on coin cells



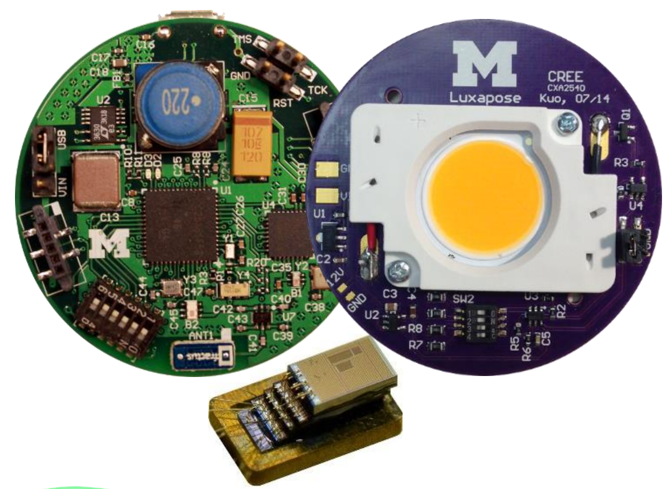
- Near-Field Communications (NFC)
  - Asymmetric backscatter technology
  - Small (mobile) readers in smartphones
  - Large (stationary) readers in infrastructure
  - New: ambient backscatter communications



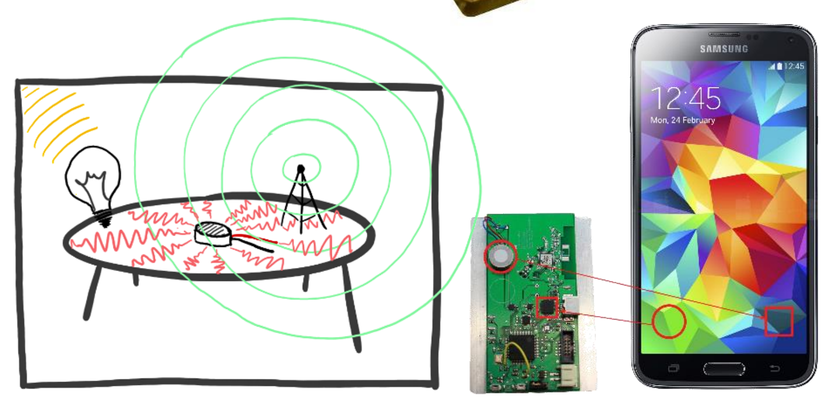
# Emerging Proximal Interfaces: Ultrasonic, Visible Light, Vibration

- Ultrasonic
  - Small, low-power, short-range
  - Supports very low-power wakeup
  - Can support pairwise ranging of nodes

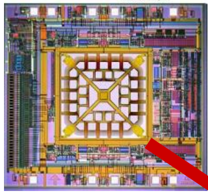
- Visible Light
  - Enabled by pervasive LEDs and cameras
  - Supports indoor localization and comms
  - Easy to modify existing LED lighting



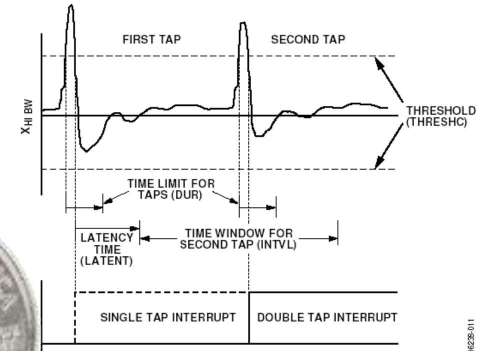

- Vibration
  - Pervasive accelerometers
  - Pervasive Vibration motors
  - Bootstrap desktop area context

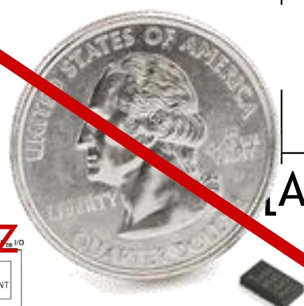


# MEMS Sensors:
# Rapidly falling price and power of accelerometers



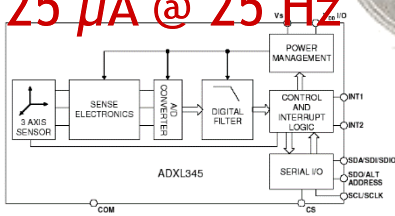
O(mA)



[Analog Devices, 2009]
ADXL345

25 $\mu$A @ 25 Hz

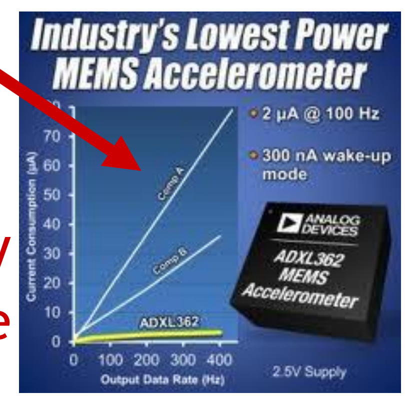



Price

Power

10 $\mu$A @ 10 Hz @ 6 bits
[ST Microelectronics, annc. 2009]


ADXL362

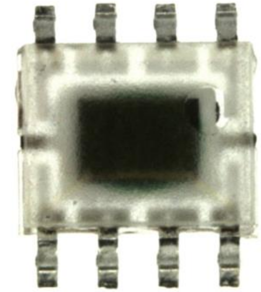



1.8 $\mu$A @ 100 Hz @ 2V
300 nA wakeup mode

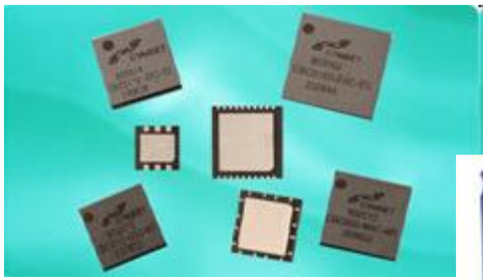
[Analog Devices, 2012]

# Energy harvesting and storage: Small doesn't mean powerless…

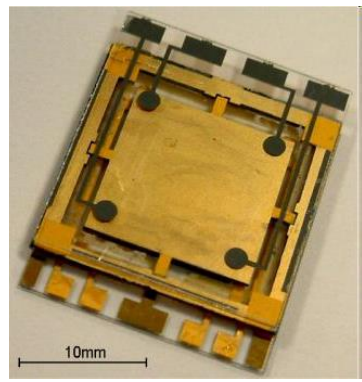

RF [Intel]



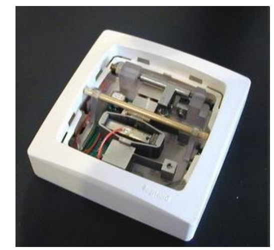
Clare Solar Cell



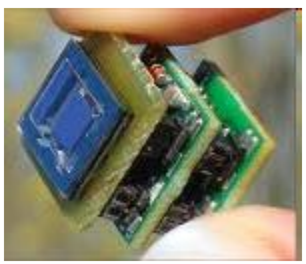

Thin-film batteries



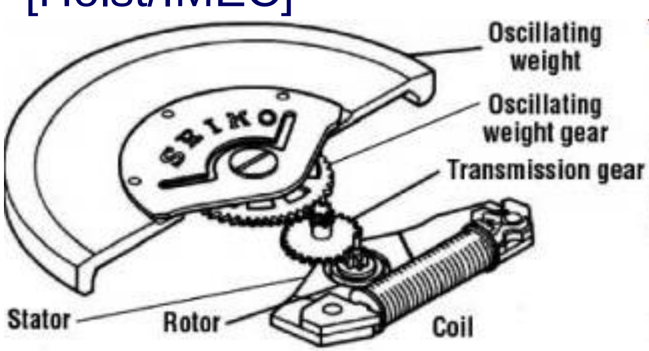

Electrostatic Energy Harvester [ICL]

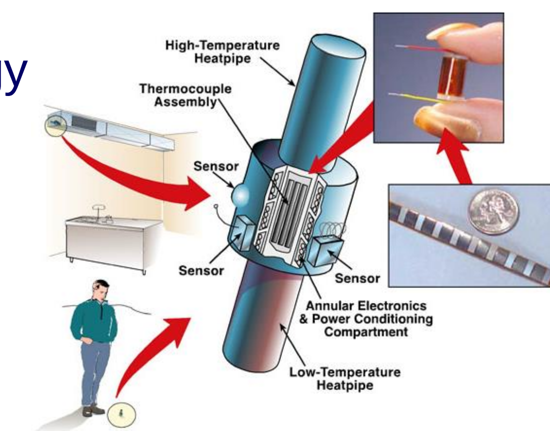


Shock Energy Harvesting CEDRAT Technologies



Piezoelectric [Holst/IMEC]

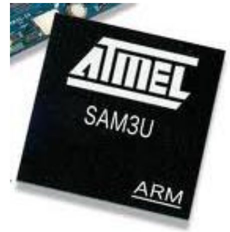



Thermoelectric Ambient Energy Harvester [PNNL]

# ARM Embedded CPU


Actel®
POWER MATTERS




ATMEL®


CYPRESS
PERFORM


DUST NETWORKS

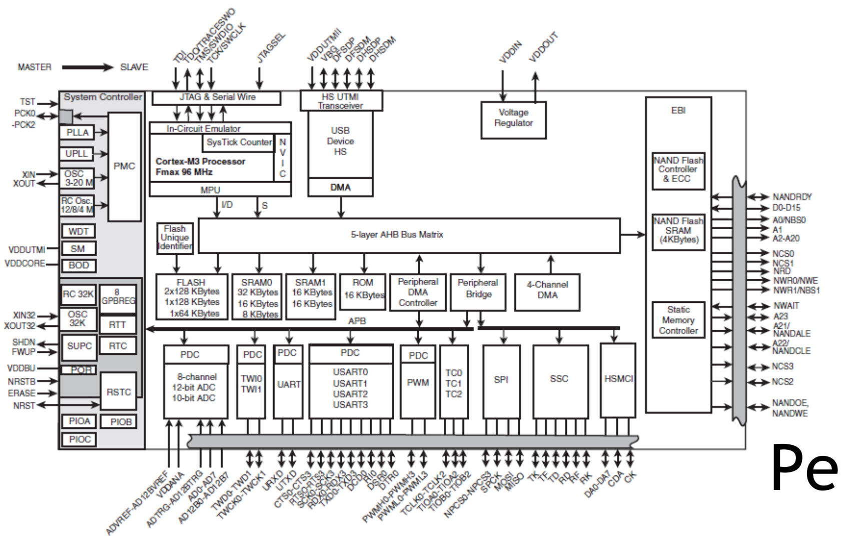

ember


NXP


ST


TEXAS INSTRUMENTS

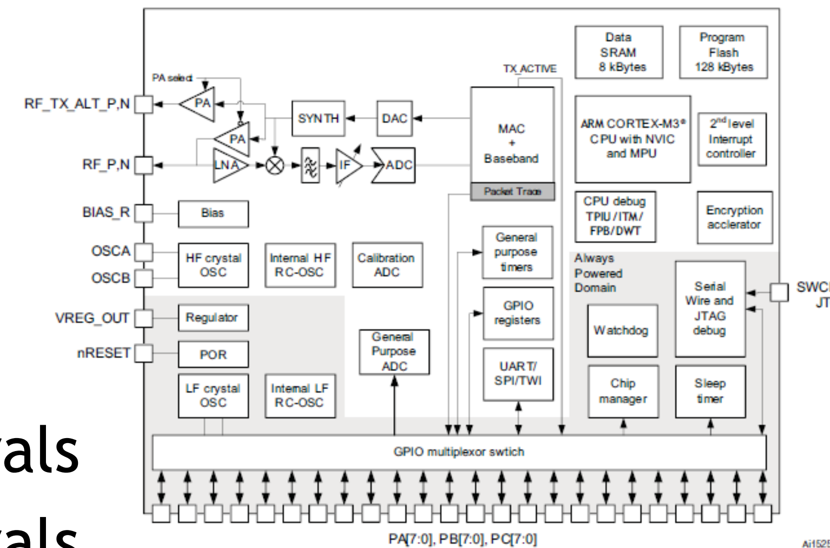
# Same ARM core with different ....

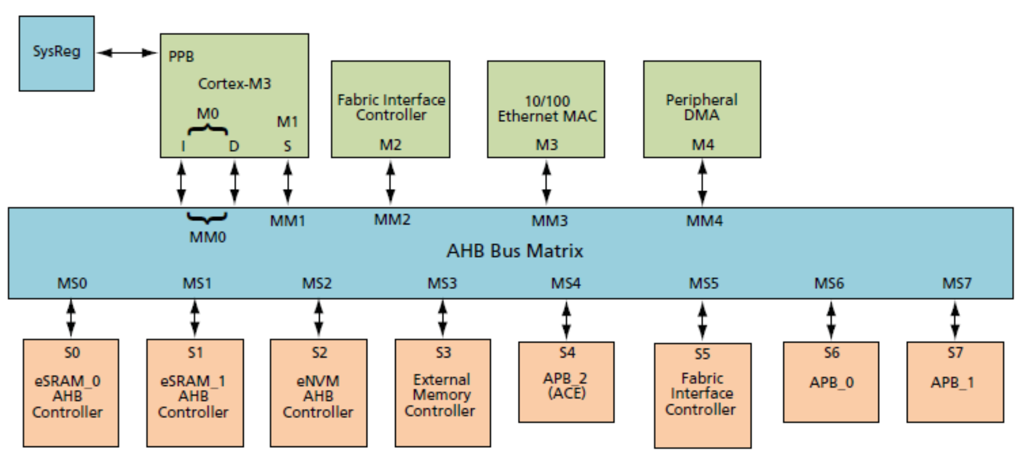



Peripherals

Peripherals

Peripherals



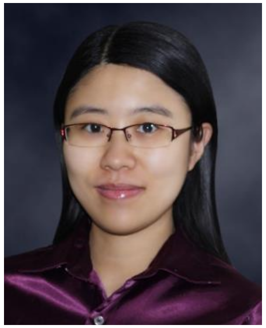



# Outline

*-Design of Intelligent SoC Robot-*

1. *Introduction*
2. *Software Algorithms for Robot*
3. *Embedded System*
4. ***Course Preview***
5. *Intelligence Robot War*

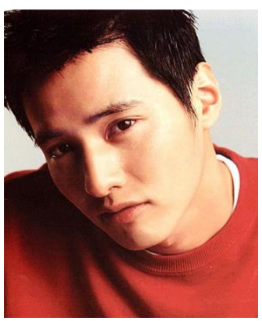
# F' 18 Instructional Staff
# (see homepage for contact info, office hours)



张沕琳
*Instructor*

Hoi-Jun Yoo
*Instructor*

Seungsu Roh
*Lab Instructor*



王学诚
*TA*

Sungpill Choi
*TA*

Jinsu Lee
*TA*

Chang-hyeon Kim
*TA*

Juhyoung Lee
*TA*

Donghyeon Han
*TA*

# Course goals

- *Learn to design* robot systems and how to think about embedded software and hardware.
- *Learn to implement* robot systems including hardware/software interfacing.
- *Design and compete your* robots for both learning and fun.

# Topics

- ARM Processor & Memory-Mapped I/O (MMIO)
  - ARM processor's characteristics, instruction set architecture.
  - How processor talk to input and output devices?
- Interface
  - Bus, UART, SPI, USB, etc.
- Field-programmable gate array (FPGA)
  - Reconfigurable hardware.
- Computer Vision (AI)
  - How to obtain information from  the visual data?
- Robot Navigation
  - What is required to complete Robot's tasks?

# Grades & Homework

| Item | Weight |
|---|---|
| Robot War Competition | 75% |
| Homework | 10% |
| Lab Report | 10% |
| Attendance | 5% |

- 1 Homework after 3 Lectures finished
  - A few problem sets about lecture materials
- 1 Lab report after each lab finished
  - Summarize what you did in lab
- Robot war competition is the most important!

# Labs

- 11 labs, 11 weeks, groups of 2
    1. ARM CPU + Memory-Mapped I/O
    2. Interface (USB, etc …)
    3. FPGA + Hardware Tools
    4. Camera and Image Processing
    5. Location and Navigation
    6. Robot Competition Preparation

- Main goals of labs are preparing robot competition

# Outline

*-Design of Intelligent SoC Robot-*

1. *Introduction*

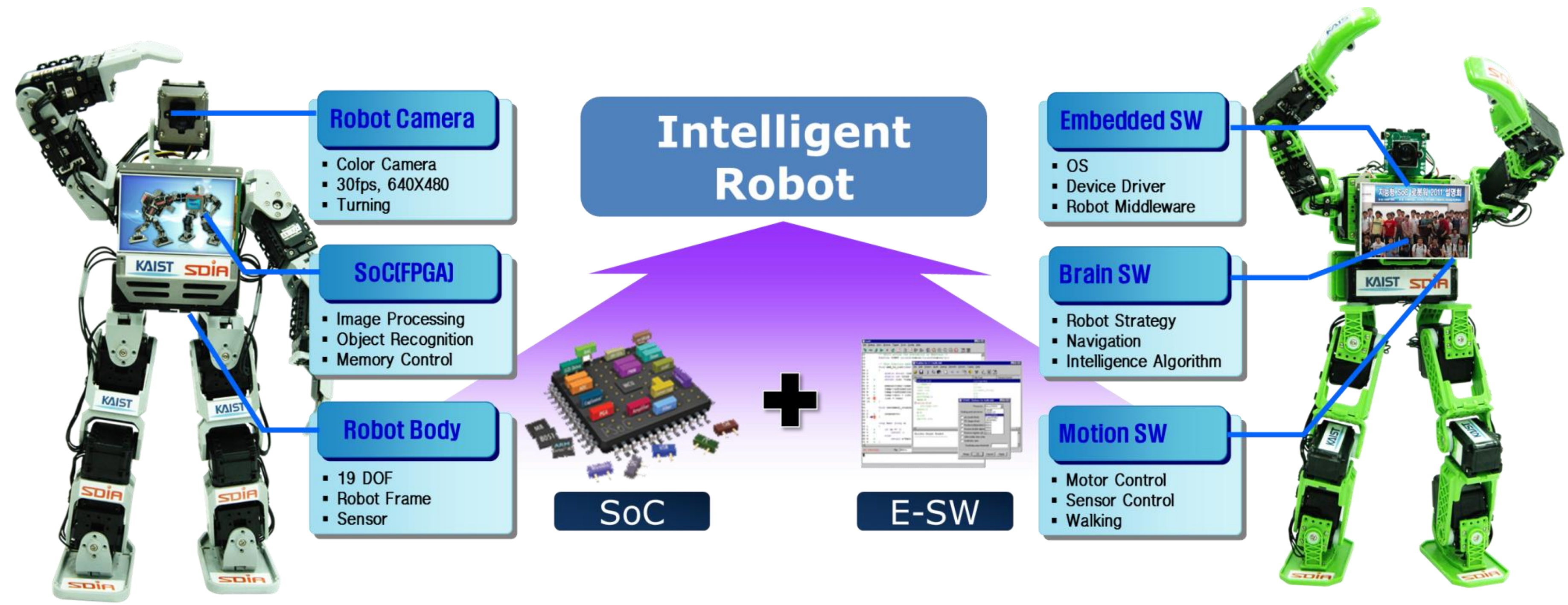
2. *Software Algorithms for Robot*

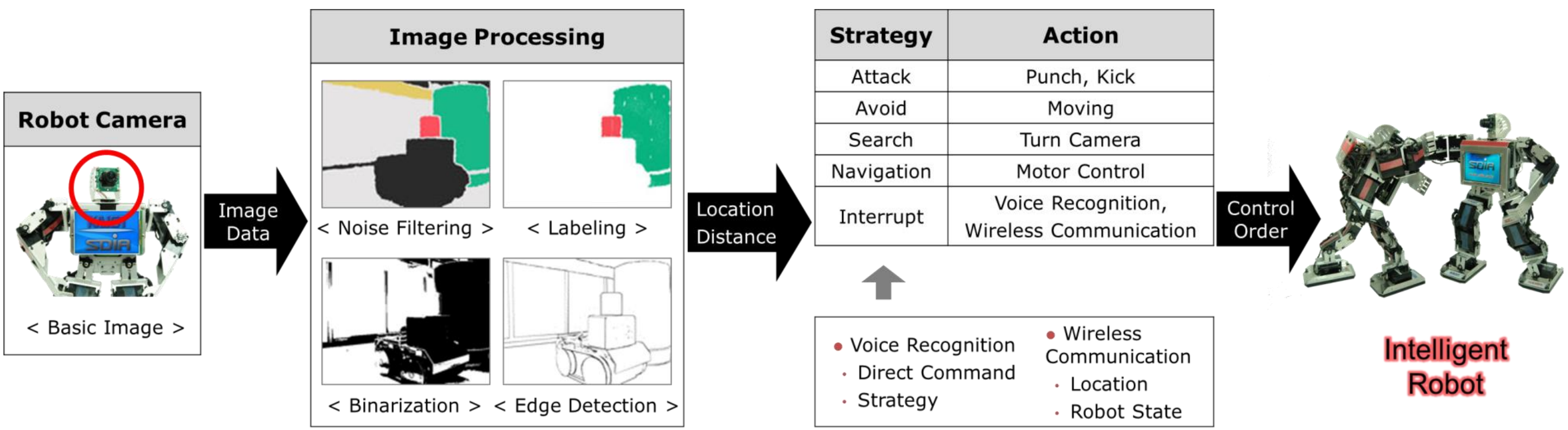
3. *Embedded System*

4. *Course Preview*

5. ***Intelligence Robot War***

# Intelligent SoC Robot System




| Strategy | Action |
|---|---|
| Attack | Punch, Kick |
| Avoid | Moving |
| Search | Turn Camera |
| Navigation | Motor Control |
| Interrupt | Voice Recognition, Wireless Communication |

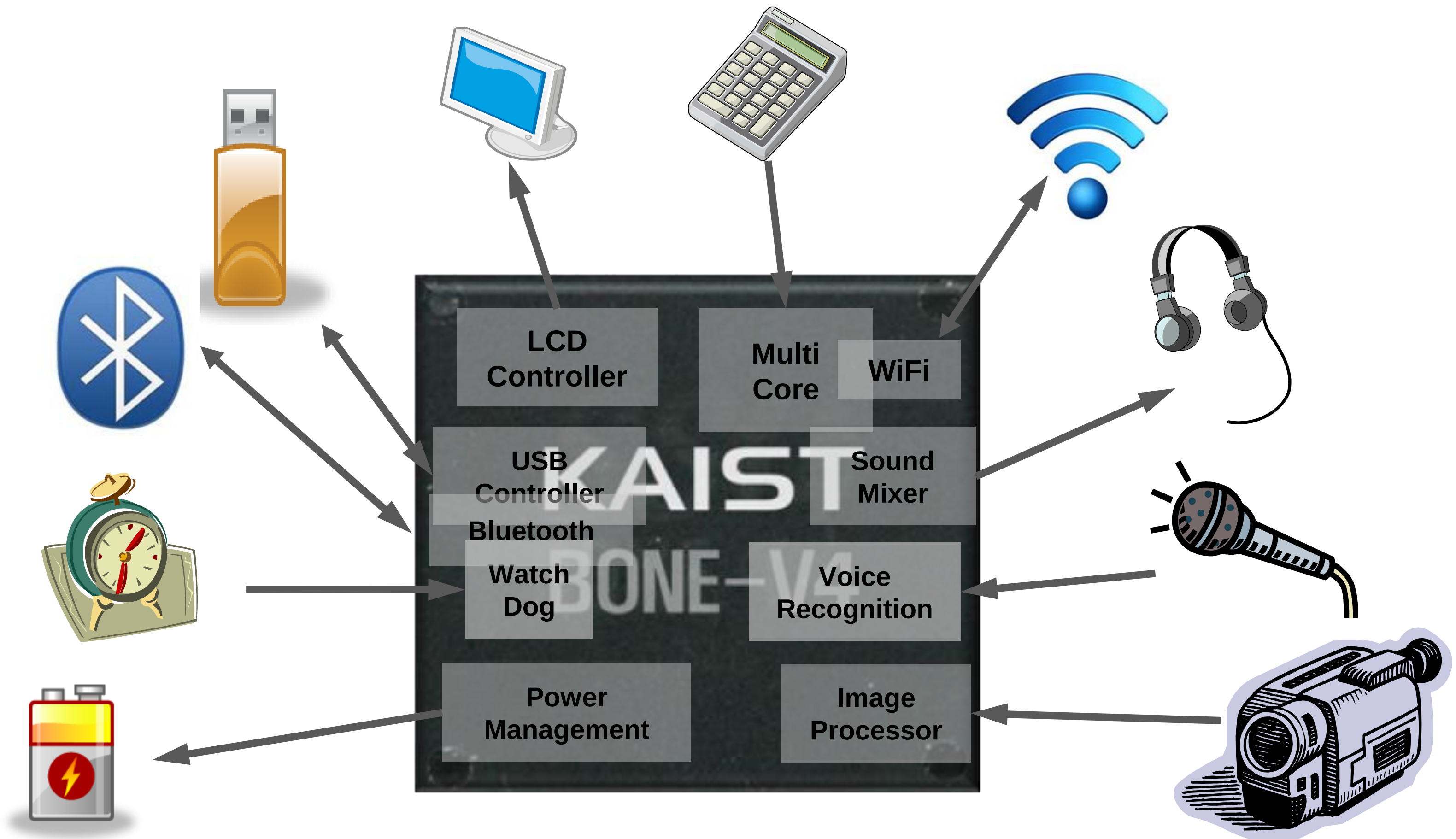
# SoC

## SoC (System On Chip)

System On Chip(SoC) is an integrated circuit that integrates all components of a computer or other electronic systems

Digital signal, Analog signal, Mixed signal, and RF is implemented on one chip

Usually used in embedded system area



# Intelligent Vision System

## Real-time Object Recognition Applications

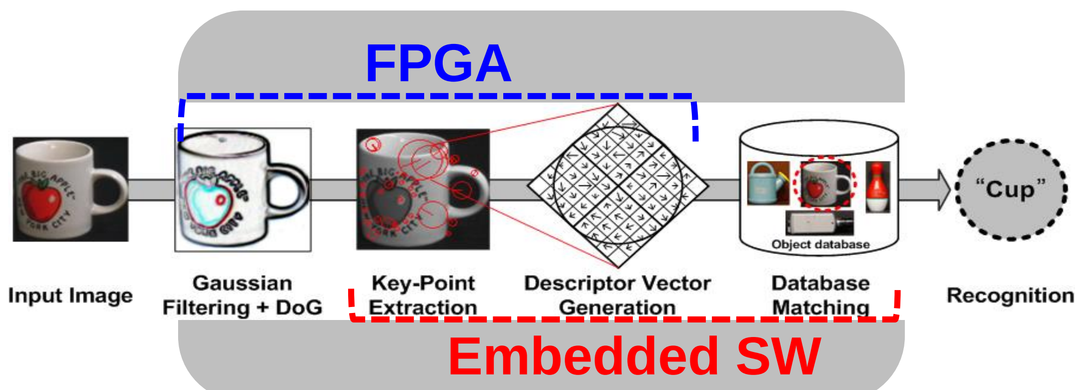
Mobile Robot Vision

Smart Phone

Surveillance System

Smart Car Vision

- Core technology in intelligent vision system
- Require large amount of image data processing and complex computation
- Cause difficulties in real time processing/recognition

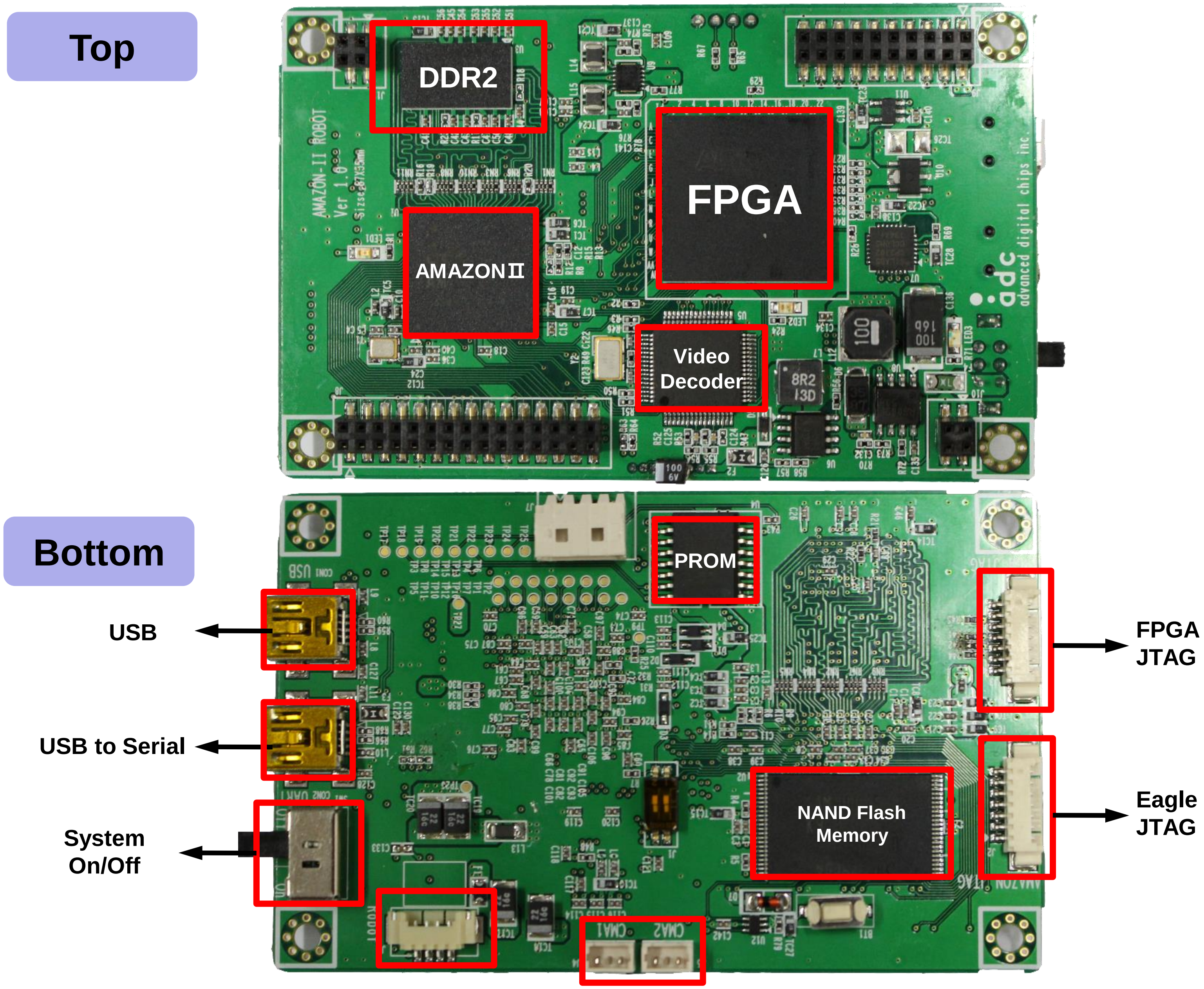



## FPGA + Embedded SW, Object Recognition Process

# Brain Board

| Item | Specification |
|---|---|
| MCU | ADChips AmazonⅡ (EISC CPU)<br>160MHz |
| Memory | DDR2 - 128MB<br>NAND Flash - 64MB |
| FPGA | Altera CycloneⅣ – EP4CE75<br>Altera EPCS64 |
| Video Decoder | SAA7111A |
| UART | 1 Port USB to Serial – PC<br>1 Port TTL Level – Robot |
| USB | 1 Port – Mini USB |
| Camera Input | 2 Port – 3Pins |
| Display | 3.5Inch TFT-LCD (320 X 480) |
| Size | 88mm X 55mm |

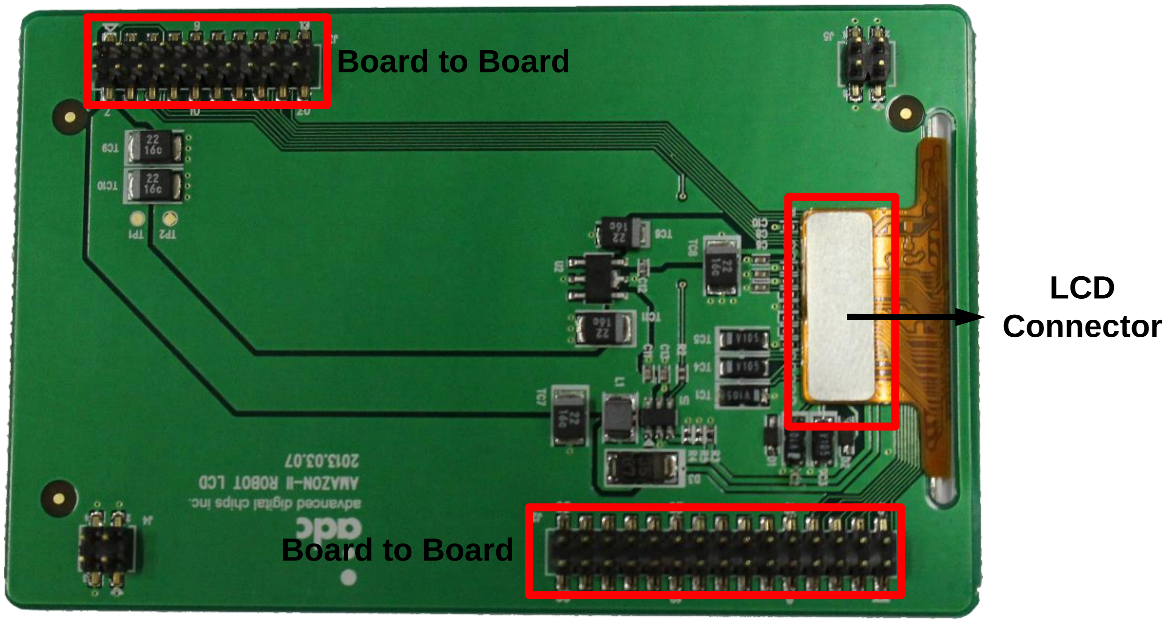
# Brain Board (Main Board)



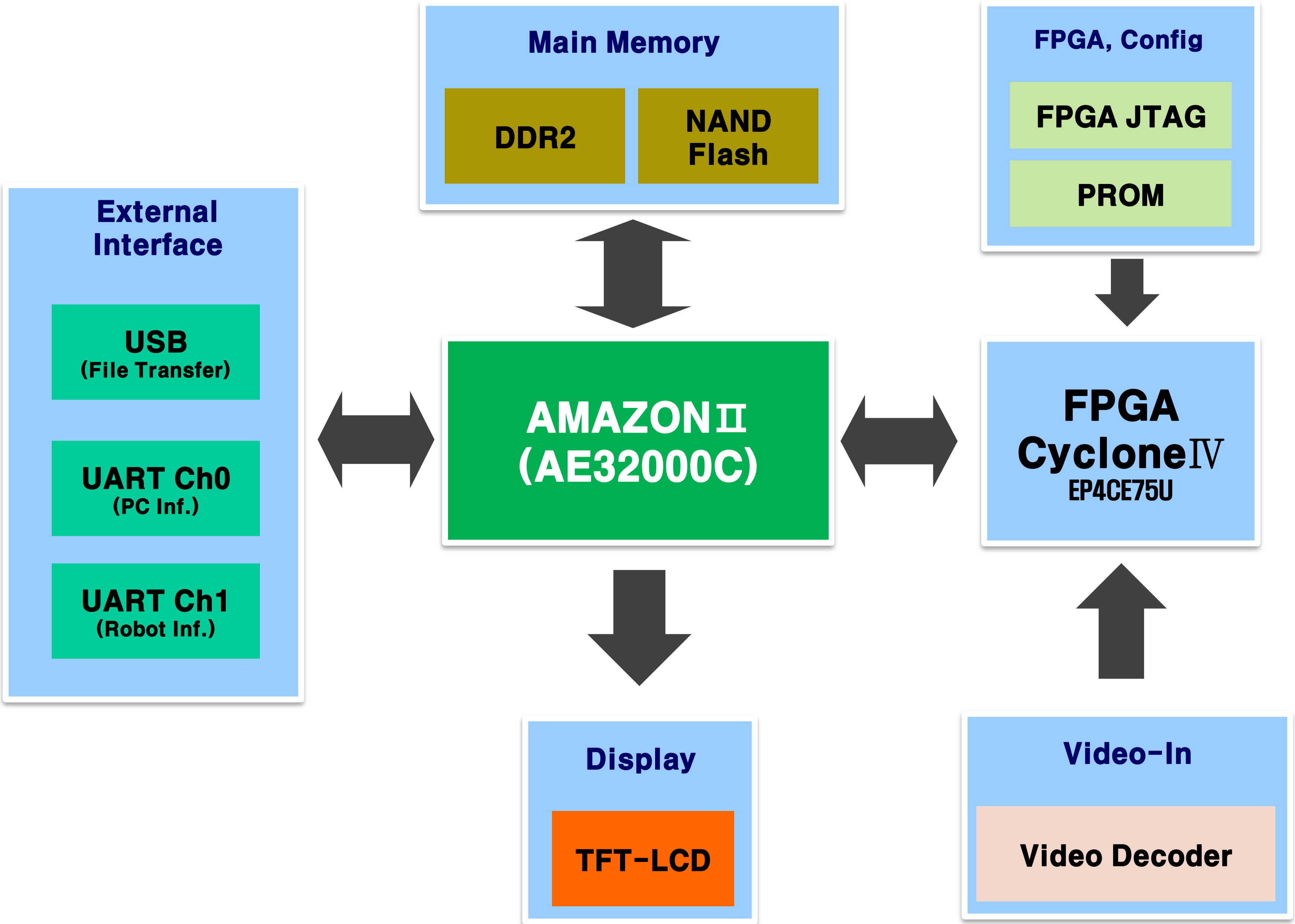
# Brain Board (LCD Board)

**Top**


**Bottom**



# Brain Board Block Diagram



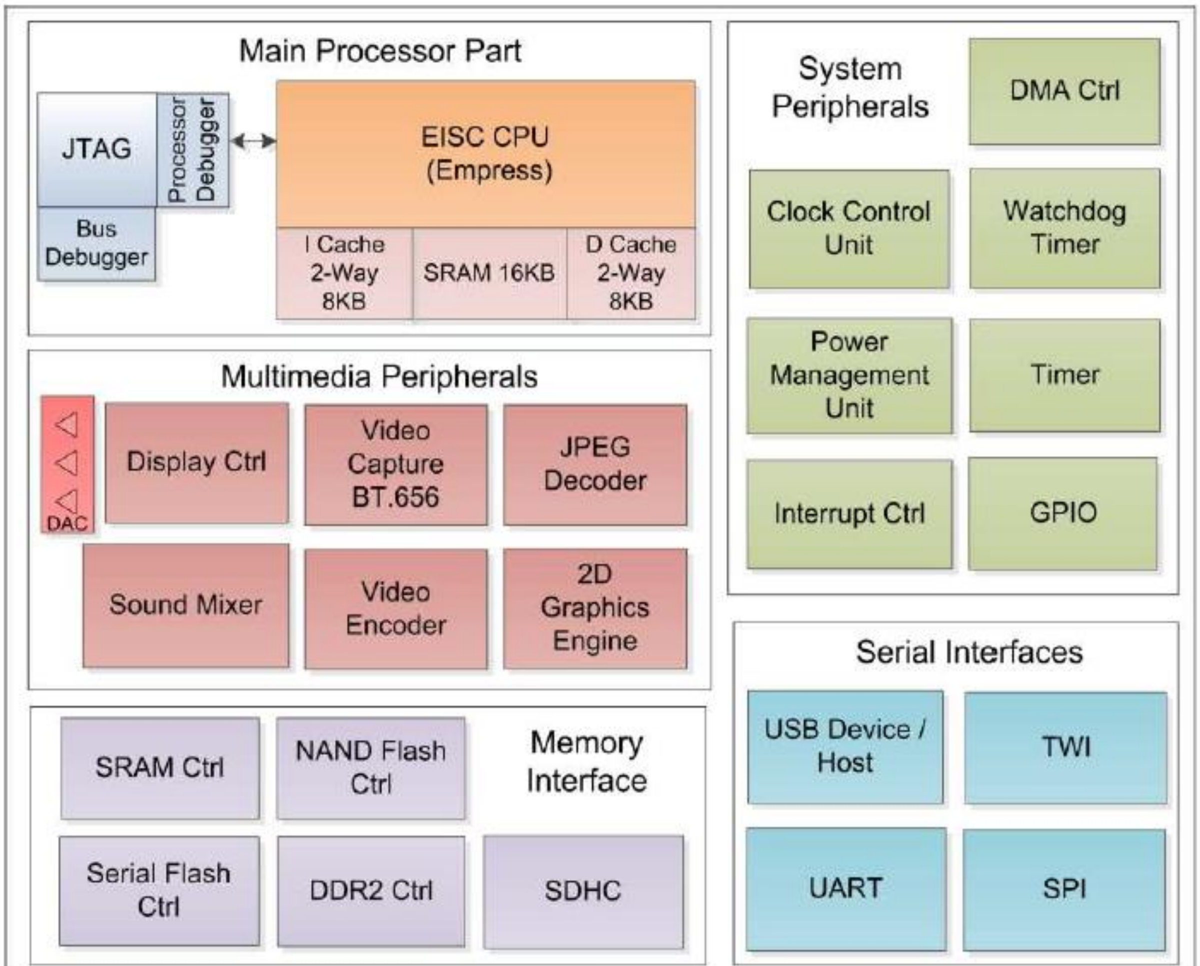
# MCU


## E-CPU

**Embedded Processor developed in Korea**

**Compensate for disadvantages of CISC and RISC**

**C/C++ compiler**



# FPGA


## FPGA “Field-Programmable Gate Array”

**Semiconductor device including logical elements and programmable internal wires**

**Slower than ASIC, cannot adapt to complex design, takes large power consumption**

**Short development time, able to remodify errors, cheap initial development price**

## HDL (Hardware Description Language)

- Computer language used to describe electronic circuits precisely
- Unlike software programming language, it can express time and synchronism, which are major features of hardware
- ABEL, AHDL, MyHDL, SystemC, VHDL, Verilog

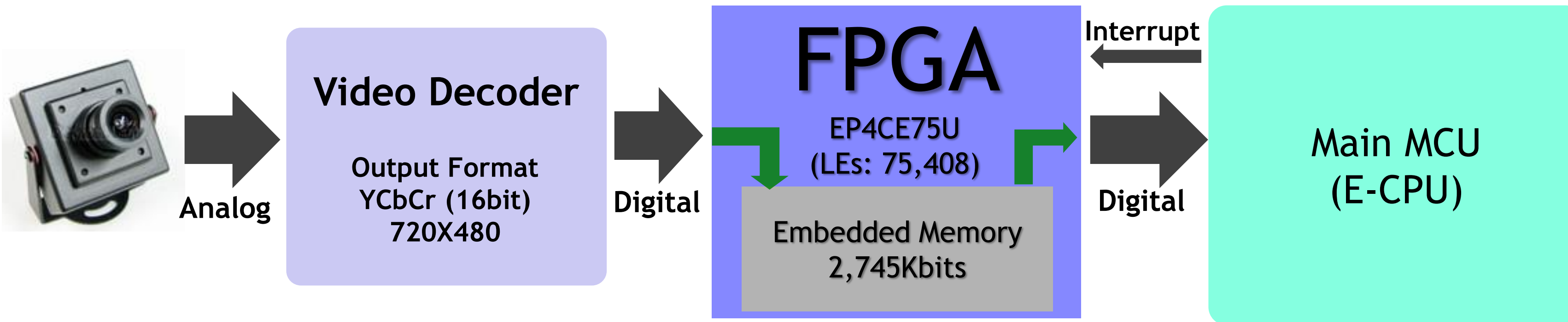
### VHDL

- Started to develop by request of Department of Defense
- 1993 standardization (IEEE 1164)
- Developed in order to improve electronic equipment used in weapon system and decrease burden on maintenance

### Verilog

- Developed by Gateway Design Automation
- Released to market as simulator product in 1985
- Similar to C language

# Image Processing




## Camera

- CCD Color Camera
- 30 frame/sec

## Video Decoder

- Decode Analog Video
- Output : 720X480, YCbCr
- 30 frame/sec

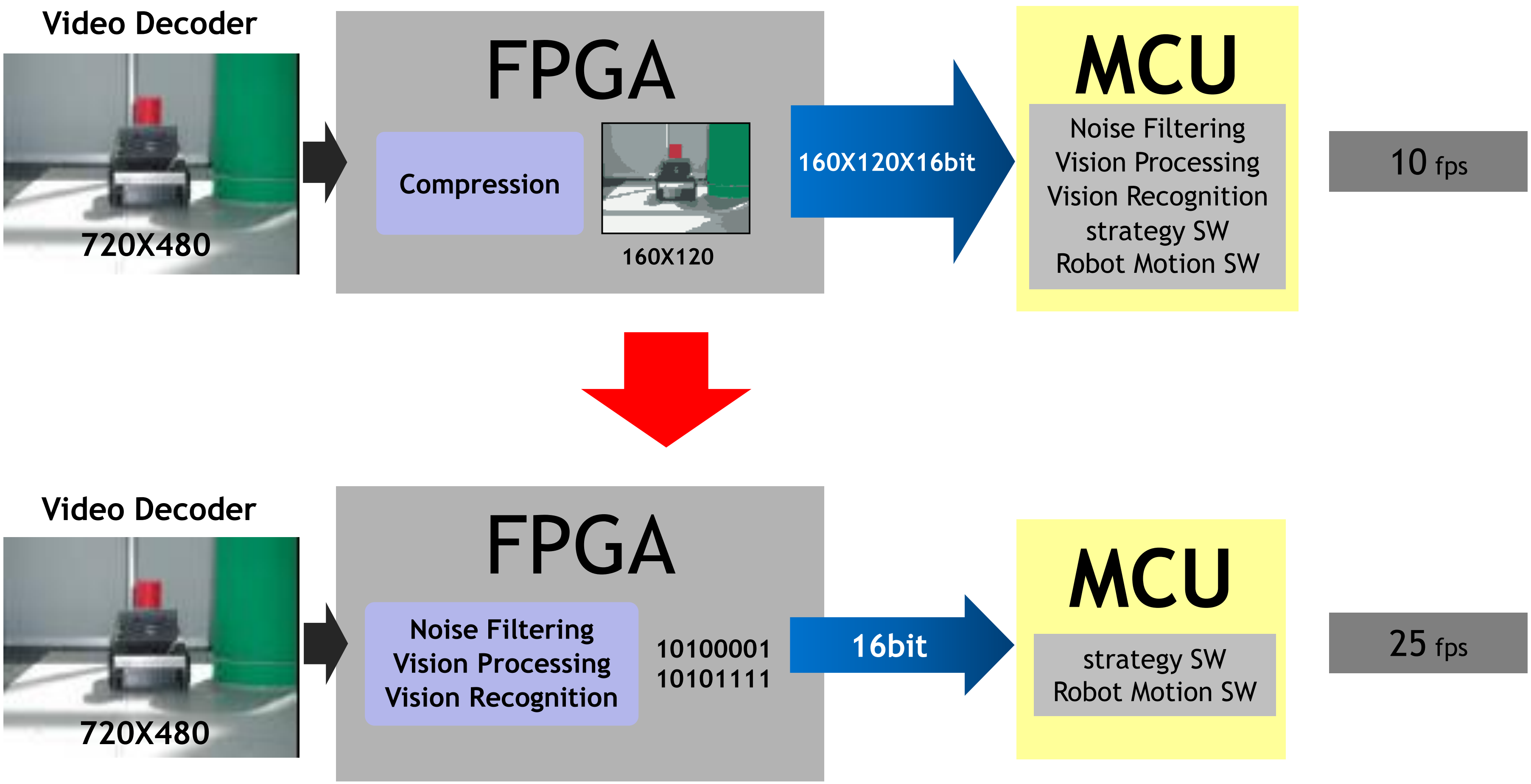
## FPGA

- Input : Image Data form Video Decoder
- Vision Processing/Recognition
- Transfer Image by request of Main MCU

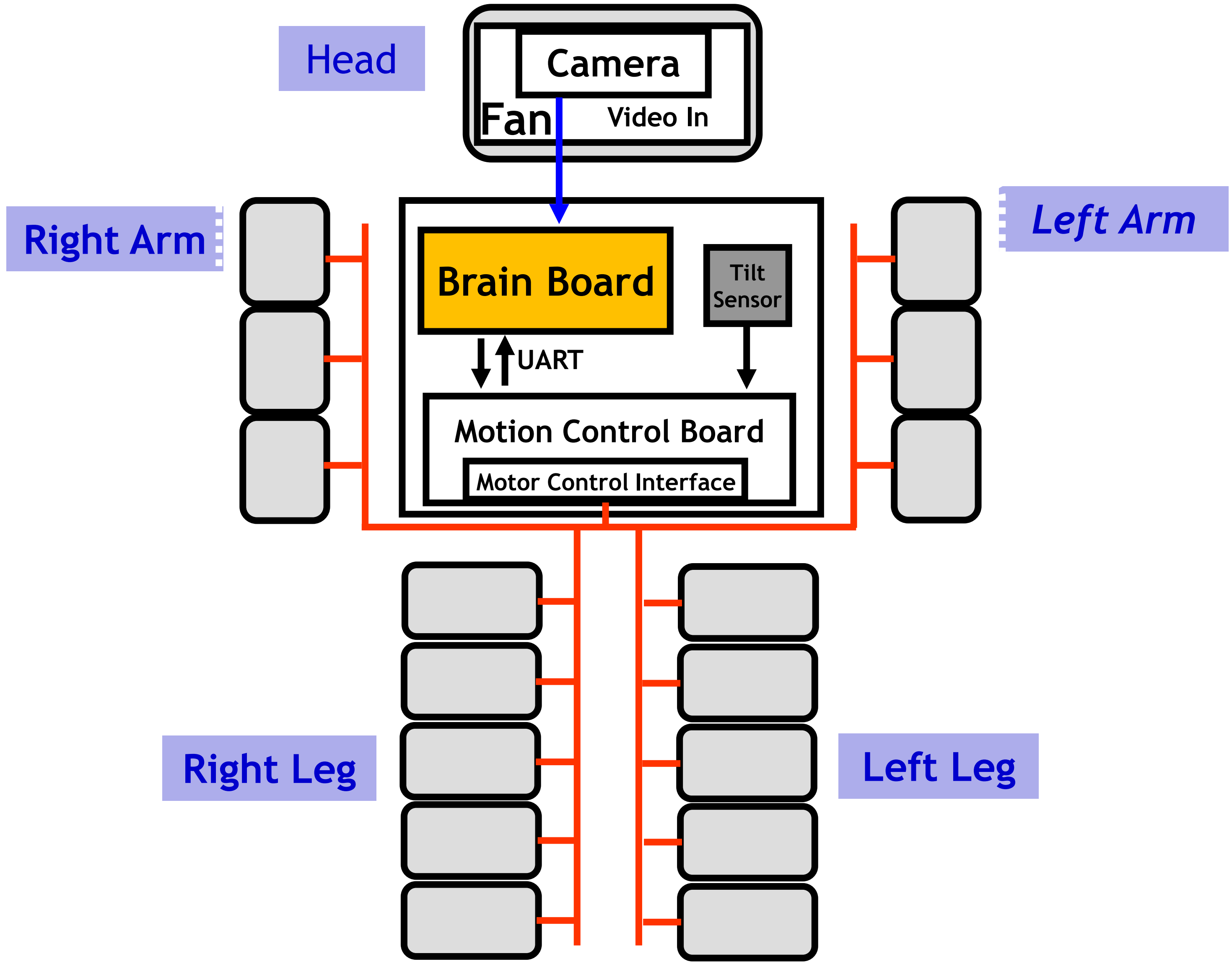
## Main MCU

- Robot Strategy
- Decision
- Robot Motion SW

# Image Processing



# SoC Robot Structure



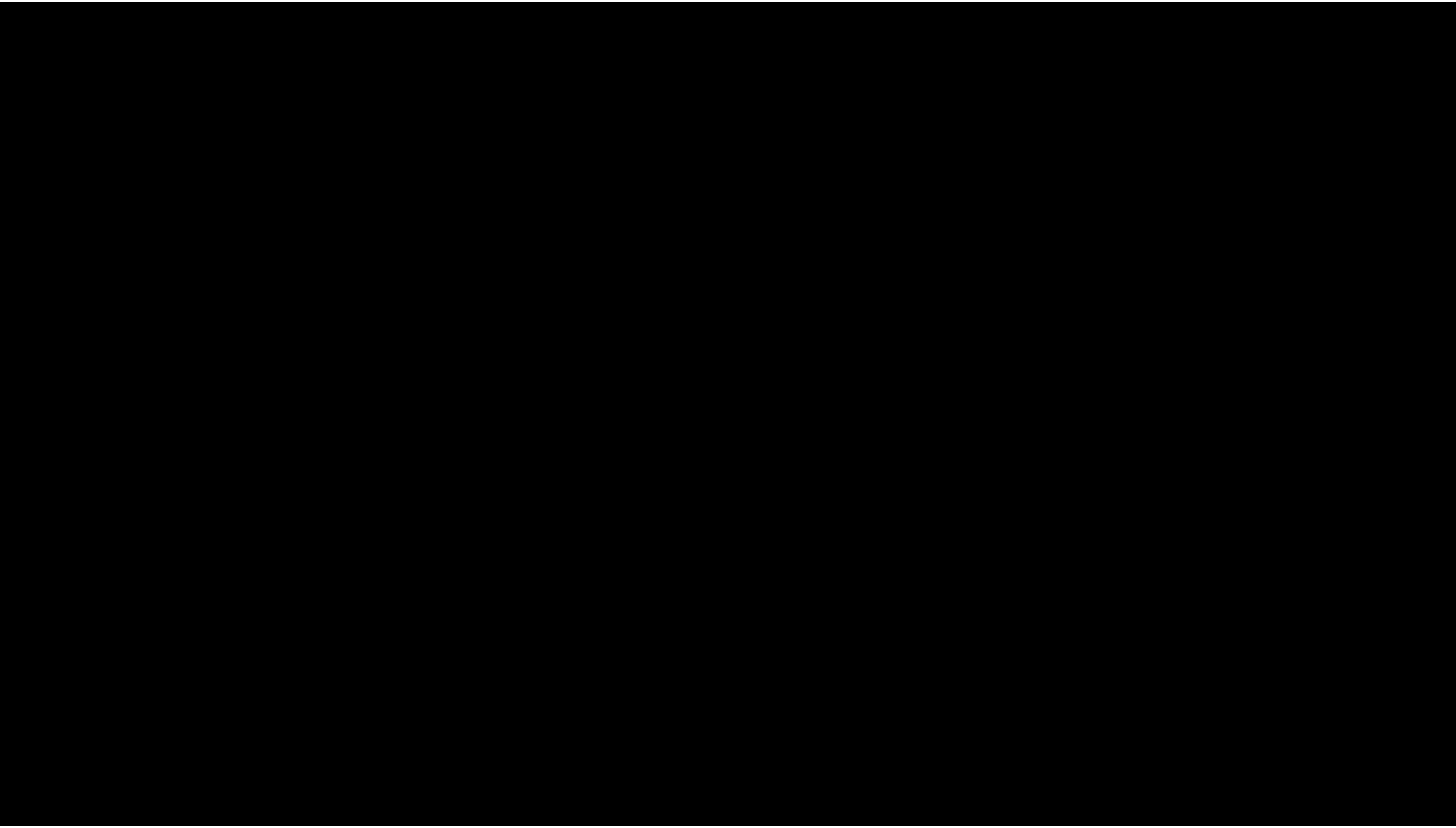
# Intelligent SoC Robot War

- Goal: learn how to build AI systems
  - By building an Intelligent Robot Controller
  - Work in teams of 2 to 3
  - You design your own robot

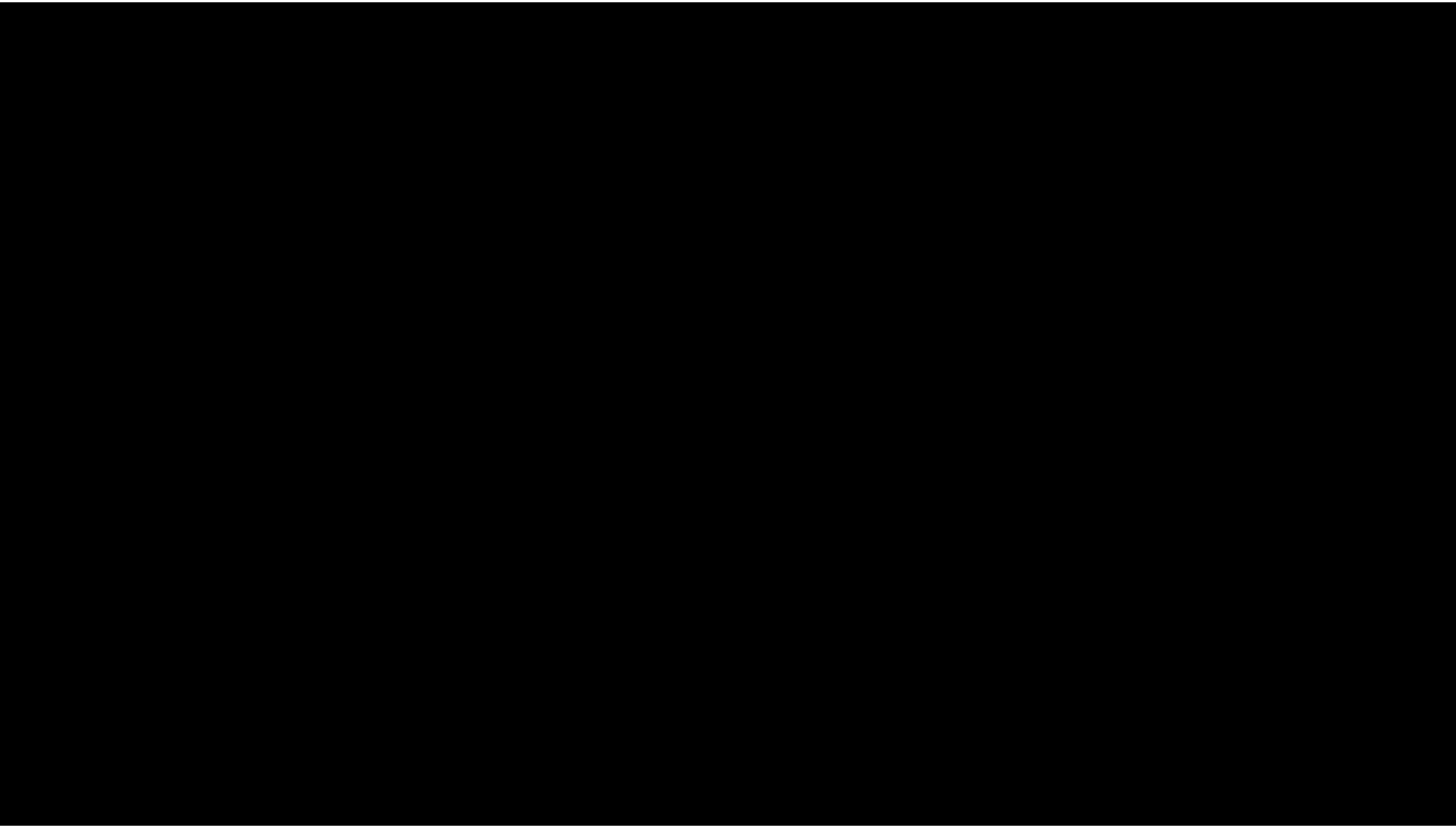
- The major focus is the Robot Competition.
  - You have to win the competition with “your robot”
  - Not only the technology, but also the fine tuning of the robot motions is important

- Important to start early and spend more time.
  - It’s better get going right away and play longer with the board and robot

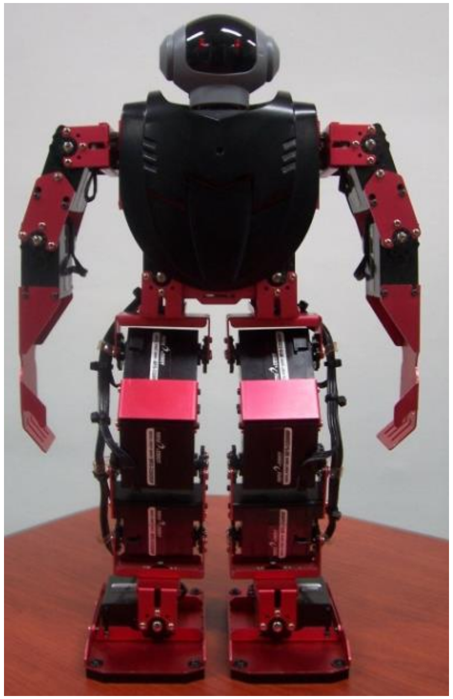
# Intelligent SoC Robot War - 1

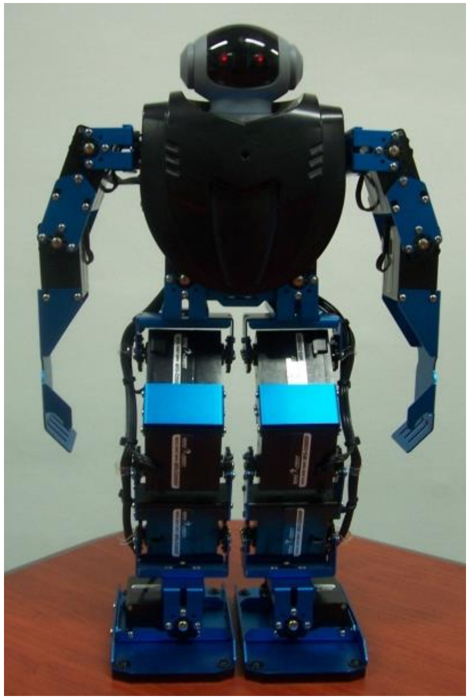


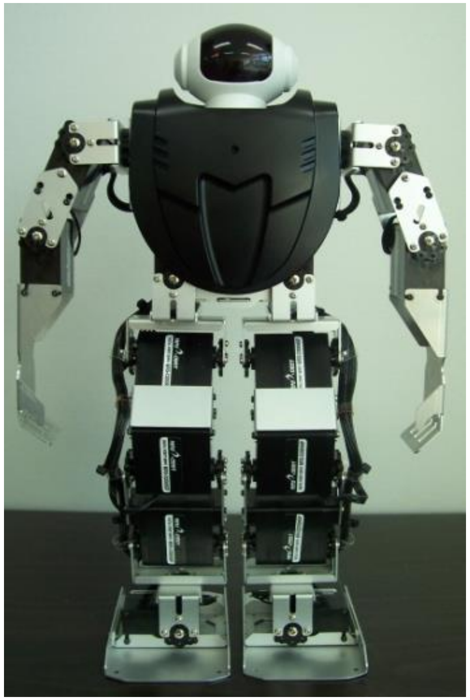
# Intelligent SoC Robot War - 2



# Example Competition Robot







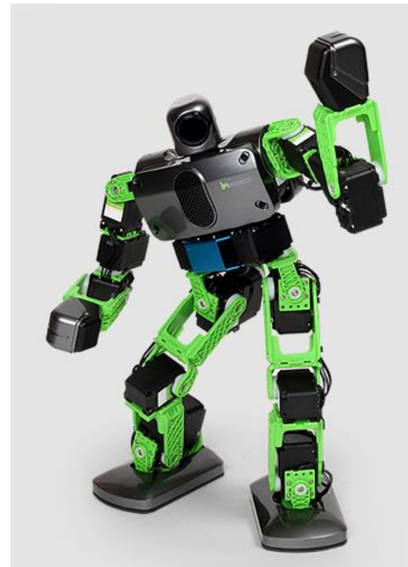
Motor: MRS-D2009SP x 17ea
Size: 310x180x90mm
Weight: 1.2kg
Controller: MR-C3024
Battery: NiMH 1000mA
Motion SW: RoboBASIC

# Example Competition Robot


Motor: Dynamic Cell x 19ea(A type)
Height: 397mm
Weight: 1.7kg
Controller: CM-510
Battery: Lithium Polymer, 1000mA
Motion SW: RoboPLUS

Motor: Dynamic Cell x 17ea
Height: 346mm
Weight: 1.6kg
Controller: CM-510
Battery: Lithium Polymer, 1000mA
Motion SW: RoboPLUS

# Example Competition Robot

Motor: DRS-0101 X 19ea

Size: 348 X 175 X 112 (mm)

Weight: 1.45Kg

Motor Controller: ATmega128

Battery: 7.4V 3,000mA Li-Po

Questions?

Comments?

Discussion?