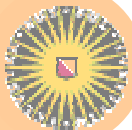


Introduction to Mobile Robotics

AIBO Programming course
2007/2008 Period 1

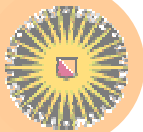
Frank Dignum

<http://www.cs.uu.nl/people/dignum>



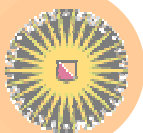
Universiteit Utrecht
dignum@cs.uu.nl

Introduction to Robotics terminology



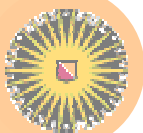
Robots and their environments

- A robot is an active, mobile, physical, artificial agent whose environment is the real physical world
- We will concentrate on **Autonomous** robots that choose their own actions based on their perception of the world
- The problem of making robot controllers is that the real world is very demanding:
- The world cannot be completely perceived, it is non-deterministic, it is dynamical, and continuous



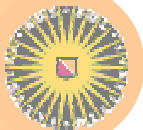
Applications of robots

- Manufacturing (repeatable tasks on production belt; car and micro-electronics industry)
- Construction industry (e.g., for shaving sheep, picking tomatoes)
- Delivery and security robots
- Unmanned vehicles (cars, airplanes, underwater)
- Tasks in dangerous environments (earthquakes, chemical substances, radio-active environments)
- Space exploration missions
- Household tasks (vacuum cleaner, lawnmower)
- Entertainment industry



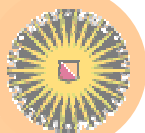
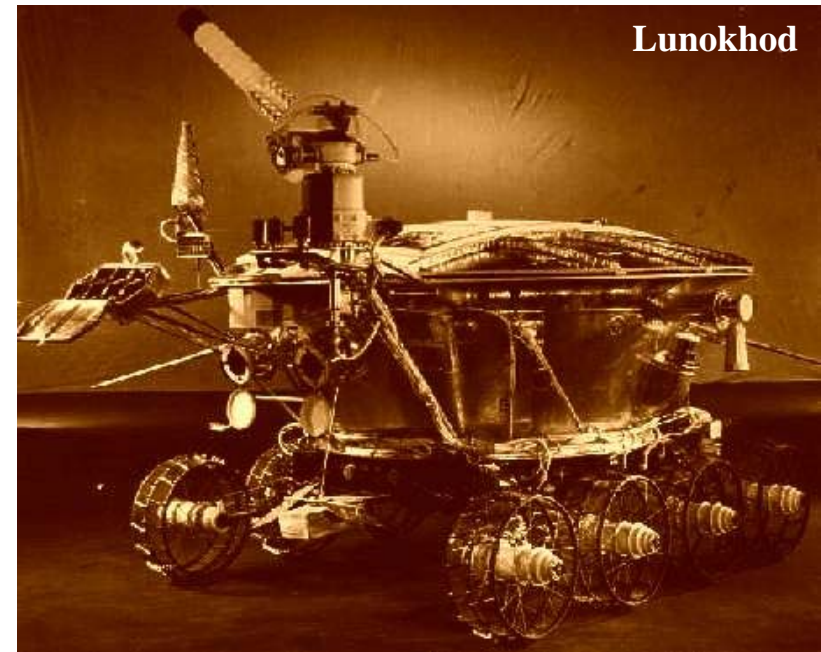
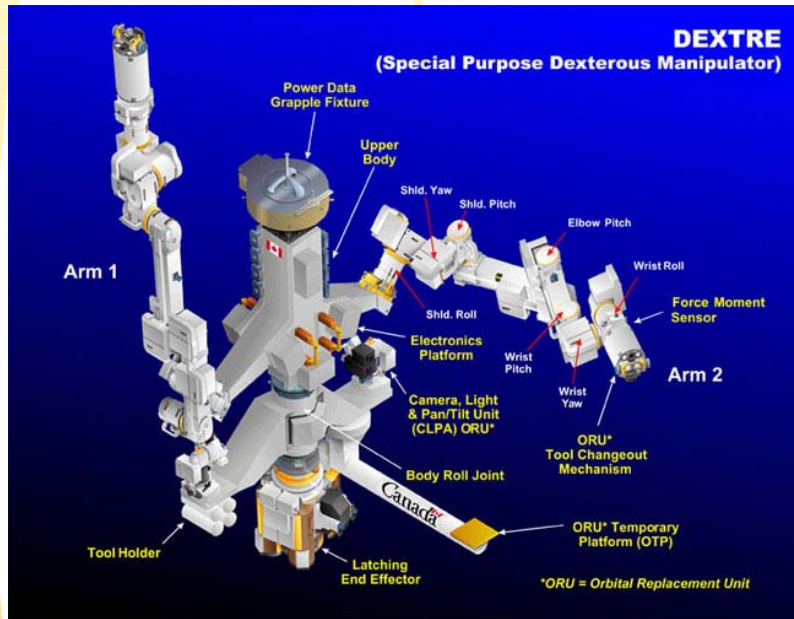
The Syllabus

- Sensors
- Vision
- Actuators
- Motion
- (Forward/Inverse) Kinematics
- Drift
- Localization
- Navigation
- Basic behaviors
- Complex behaviors
- Multi-robot behaviors
- Inertia
- Torque
- Compass
- Joint
- Bumpers
- Landmark
- Geometric Map
-



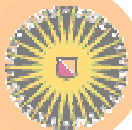
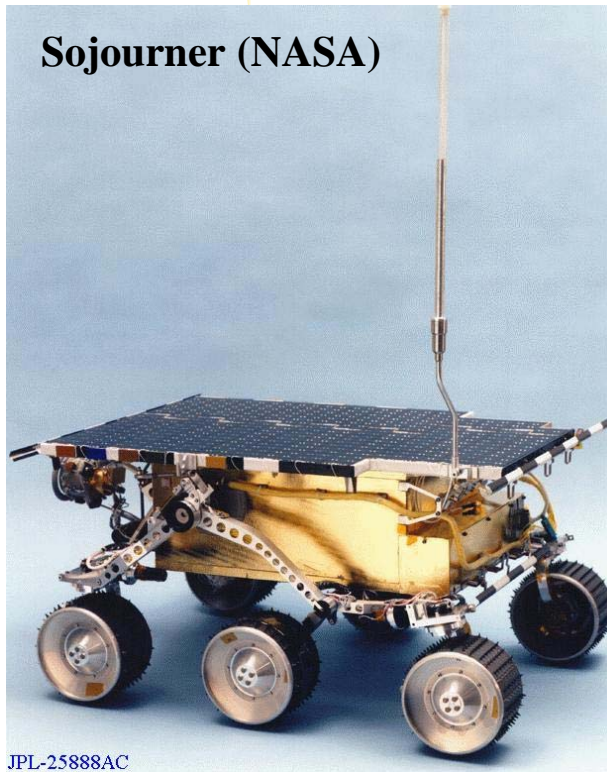
Types of robots (I)

- Static Robots vs Mobile Robots



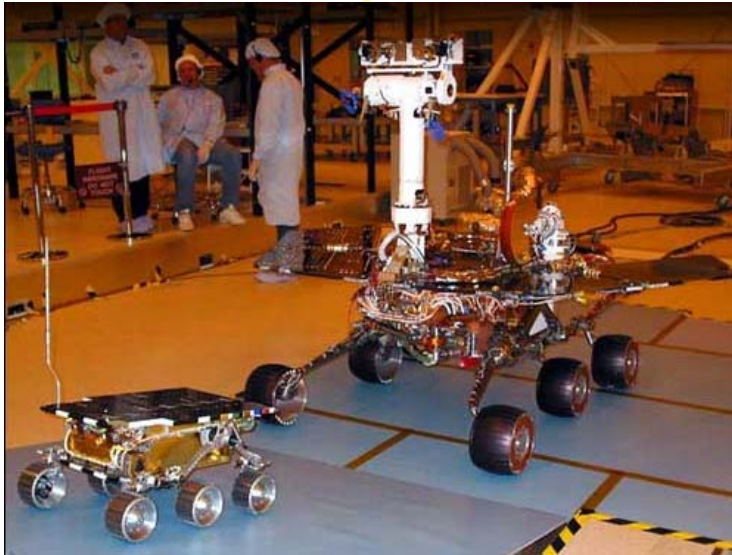
Types of robots (II)

- Wheeled Robots VS Legged Robots



Types of robots (III)

- Robots,

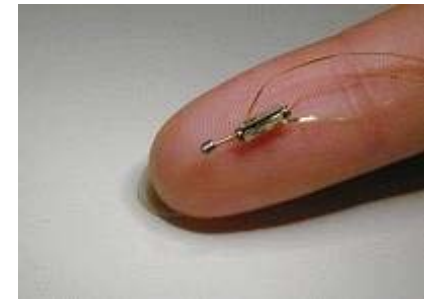


- Microbots,

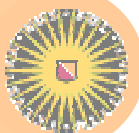
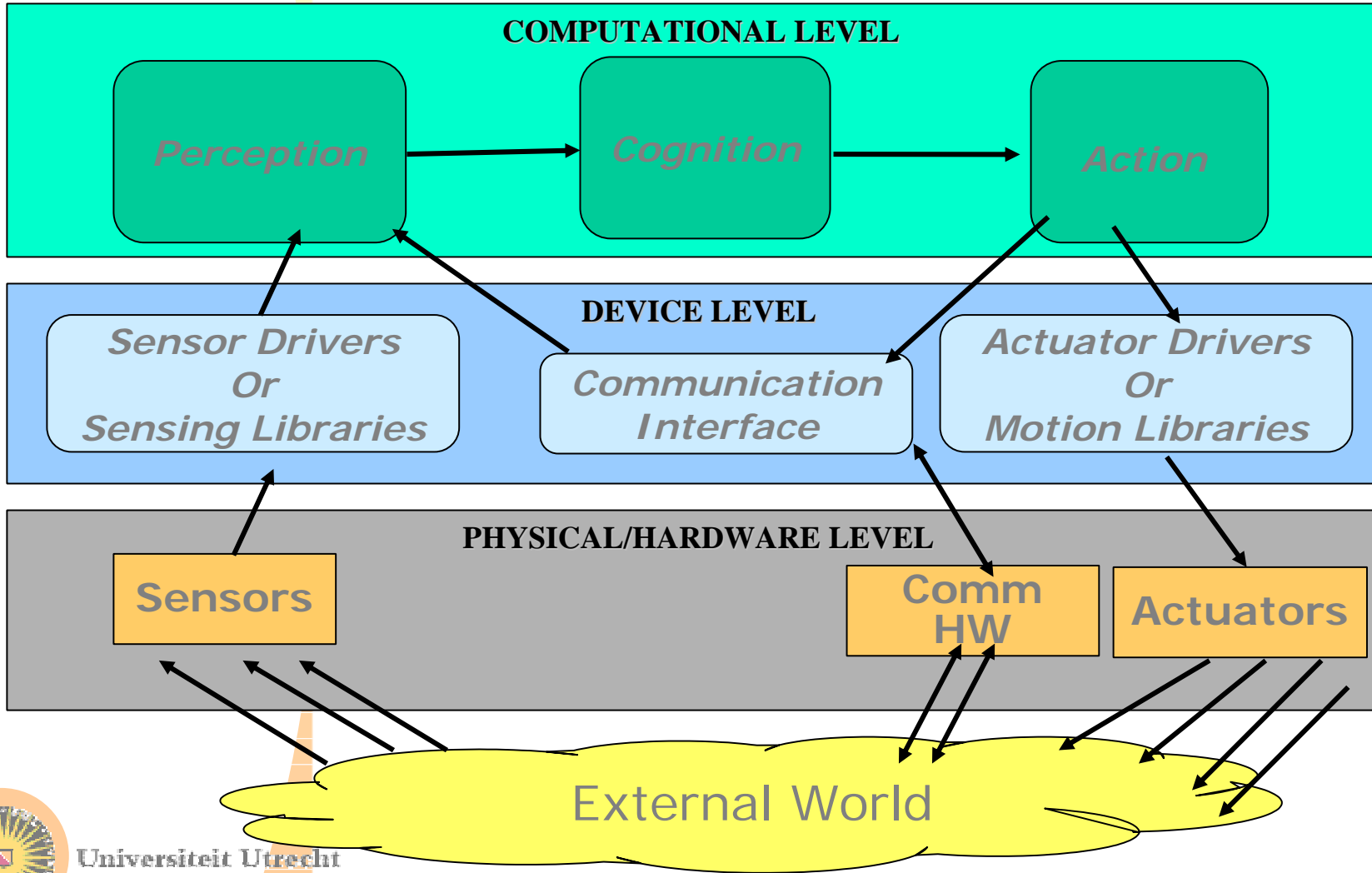


- Small, cheap robots,
- cheap sensors (no sonar or laser)

- Nanobots



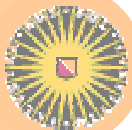
Levels of abstraction



Intelligent Robot

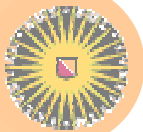
Tasks

- Perception
 - sensing, modelling of the world
 - Communication (*listening*)
- Cognition
 - behaviours, action selection, planning, learning
 - multi-robot coordination, teamwork
 - response to opponent, multi-agent learning
- Action
 - motion, navigation, obstacle avoidance
 - Communication (*telling*)



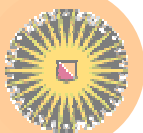
Perception

- **Non-visual sensors**
- **Vision (segmentation, colour)**
- **Localization**



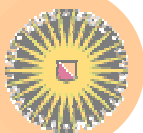
Perception: Non-visual sensors

- Laser range finder (scanner)
- Sonar (SOund NAvigation and Ranging)
- Proprioception (what are the joint-positions?)
- Odometry (measures wheel rotations)
- Force Sensors
- Touch Sensors (exist also in an AIBO)
- Infrared sensors



Perception: Vision

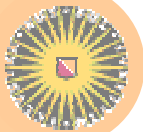
- Vision is a way to relate measurement to scene structure
 - Our human environments are shaped to be navigated by vision
 - E.g., road lines
 - Problem: vision technology is not very well-developed
 - Recognition of shapes, forms,
 - Solution: in most of cases, we don't need full recognition
 - Use our knowledge of the domain to ease vision
 - E.g.: Green space in a soccer field means free (void) space.
- Two kinds of vision:
 - Passive vision: static cameras
 - Processing of snapshots
 - Active vision: the camera moves.
 - Intricate relation between camera and the environment



Perception: Vision

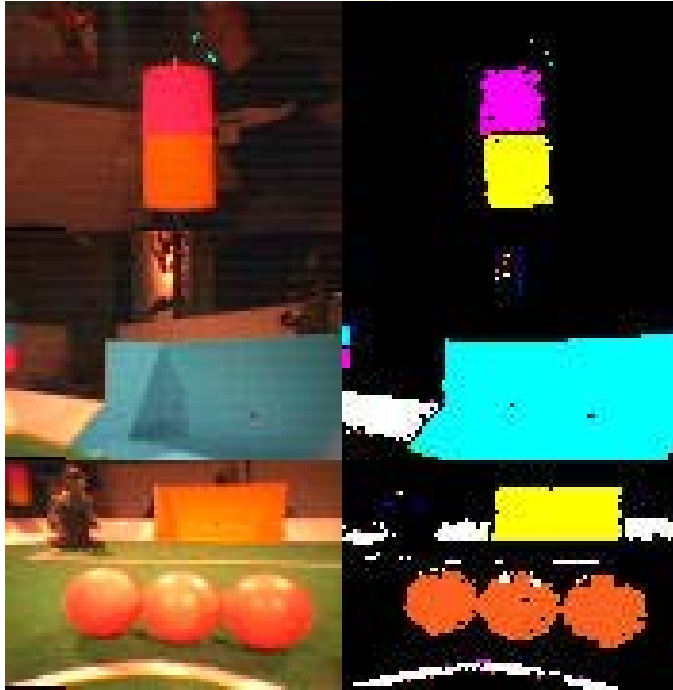
Active Vision

- Important geometric relation between the camera and the environment
 - Movement of the camera should produce an (expected) change in the image
- Useful to increase the visual information of an item
 - Move to avoid another object that blocks the vision
 - Move to have another viewpoint of the object, and ease recognition
 - Move to measure distances by comparison of images
 - An improvement: Stereo Vision
- Active Vision is highly sensible to calibration
 - Geometric calibration
 - Color calibration



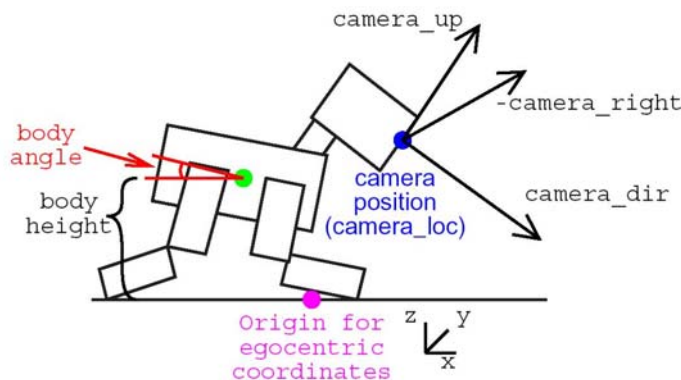
Perception: Vision

Active Vision



- Color calibration
 - Identify the colors of landmarks and important objects
 - Adaptation to local light condition
 - Saturation of color
 - Colored blobs identified as objects
 - Problem: threshold selection

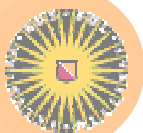
- Geometric calibration
 - Position of the camera related to the floor
 - At least 3 coordinate systems
 - Egocentric coordinates
 - Camera coordinates
 - Translation matrix counting intermediate joints



Perception: Vision

Image Segmentation

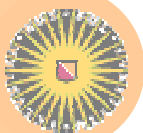
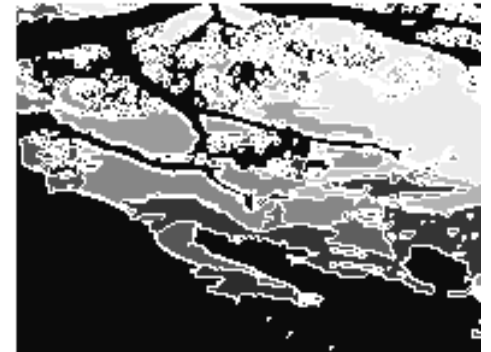
- Sort pixels into classes
- Obstacle:
 - Red robot
 - Blue robot
 - White wall
 - Yellow goal
 - Cyan goal
 - Unknown color
- Free space :
 - Green field
- Undefined occupancy:
 - Orange ball
 - White line



Perception: Vision

Image Segmentation by region growing

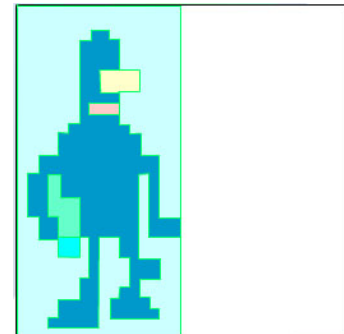
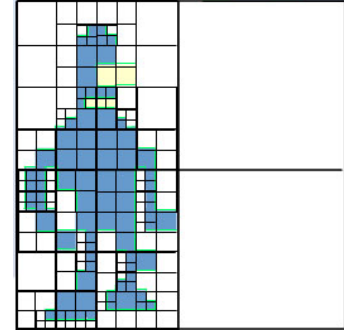
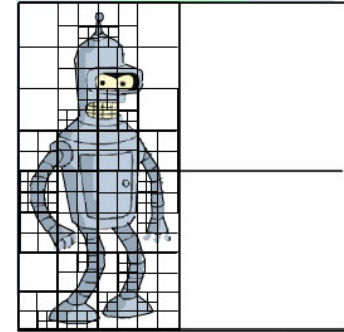
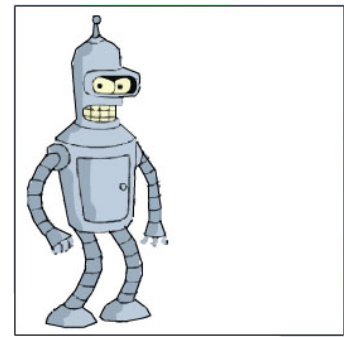
- Start with a single pixel p and wish to expand from that *seed pixel* to fill a coherent region.
- Define a similarity measure $\mathcal{S}(i, j)$ such that it produces a high result if pixels i and j are similar
- Add pixel q to neighbouring pixel p 's region iff $\mathcal{S}(p, q) > T$ for some threshold T .
- We can then proceed to the other neighbors of p and do likewise, and then those of q
- *Problems.*
 - highly sensible to the selection of the seed and the Threshold
 - computationally expensive because the merging process starts from small initial regions (individual points).



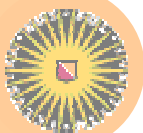
Perception: Vision

Image Segmentation by Split and Merge

- **Split** the image. Start by considering the entire image as one region.
 - If the entire region is coherent (i.e., if all pixels in the region have sufficient similarity), leave it unmodified.
 - If the region is not sufficiently coherent, split it into four quadrants and recursively apply these steps to each new region.
- The “splitting” phase builds a quadtree
 - several adjacent squares of varying sizes might have similar characteristics.
- **Merge** these squares into larger coherent regions from the bottom up.
 - Since it starts with regions (hopefully) larger than single pixels, this method is more efficient.

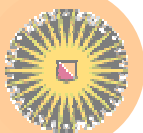


(Example by C. Urdiales)

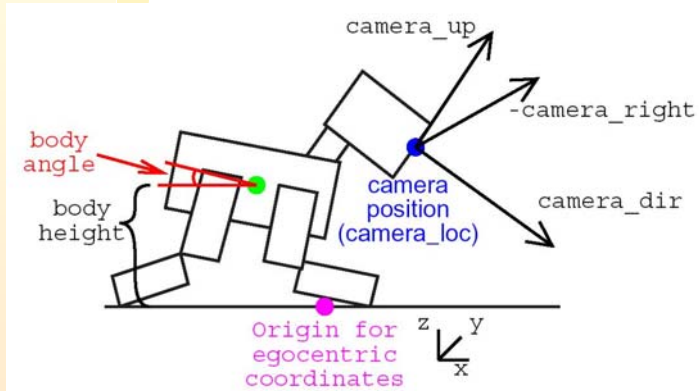


Perception: Localization

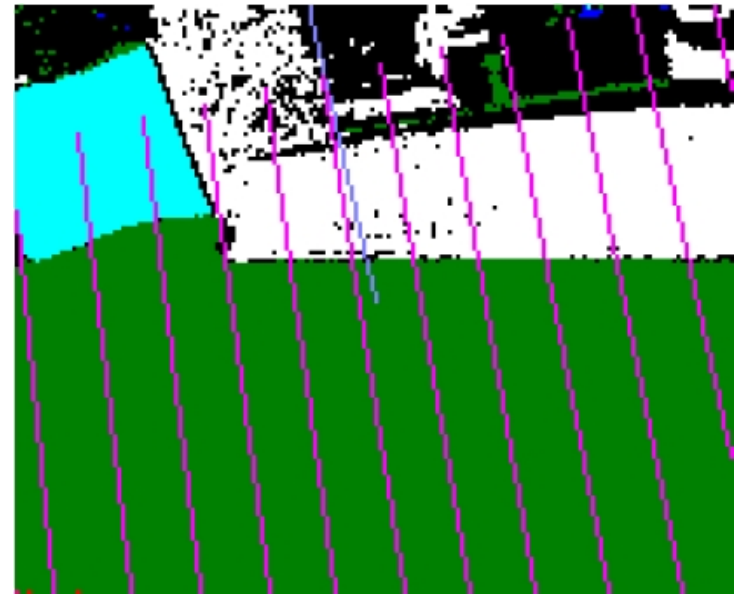
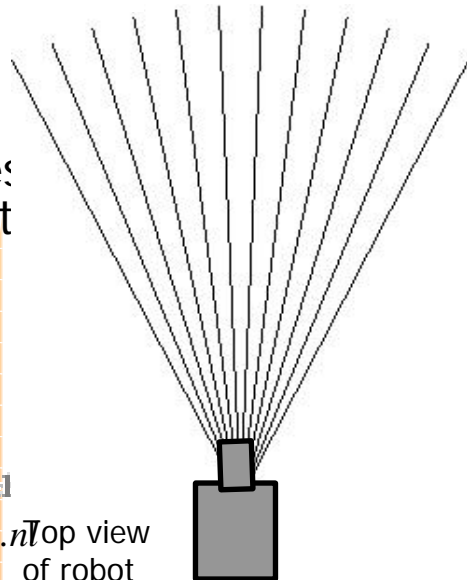
- *Where am I?*
- Given a map, determine the robot's location
 - Landmark locations are known, but the robot's position is not
 - From sensor readings, the robot must be able to infer its most likely position on the field
 - Example : where are the AIBOs on the soccer field?



Scanning Image for Objects



Scanlines projected from origin for egocentric coordinates in 5 degree increment

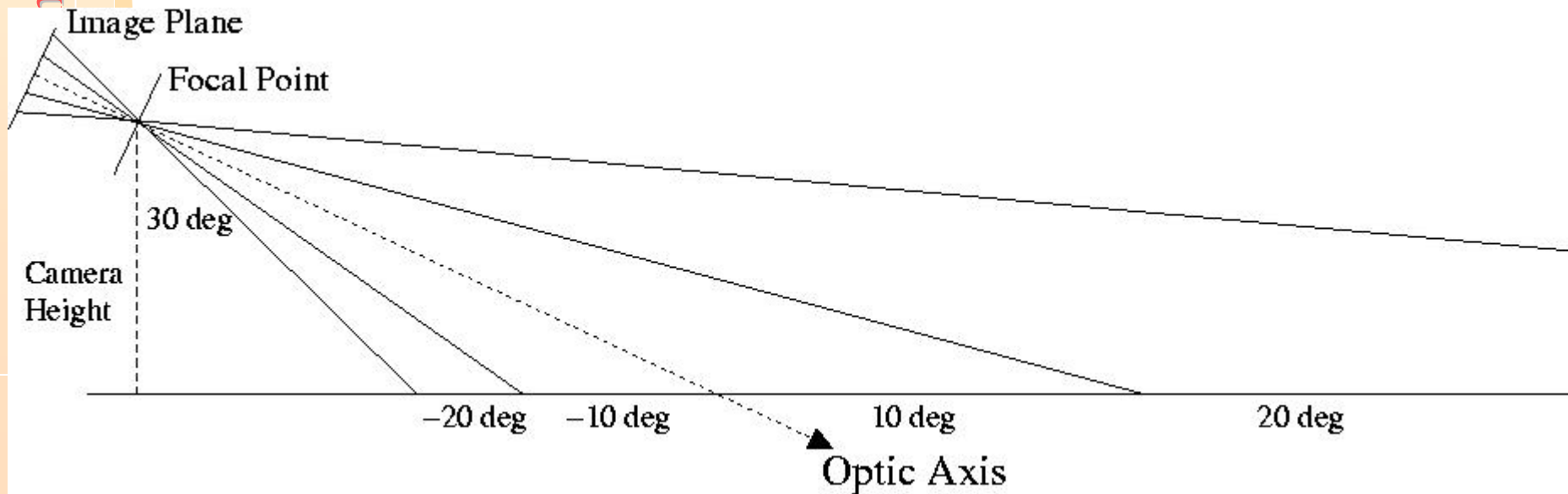


Scanlines projected onto RLE image



Measuring Distances with the AIBO's Camera

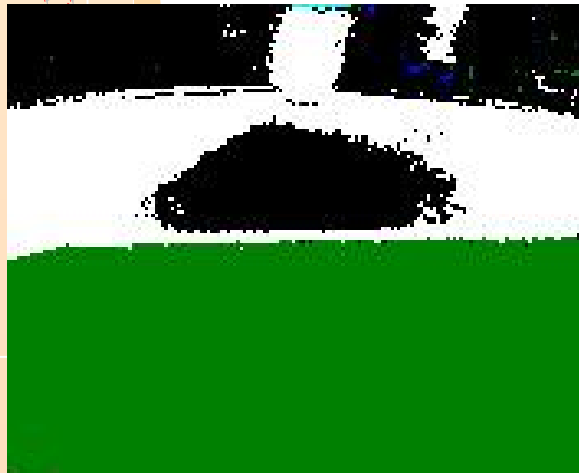
- Assume a common ground plane
- Assume objects are on the ground plane
 - Elevated objects will appear further away
 - Increased distance causes loss of resolution



Identifying Objects in Image

- Along each scanline:
 - Identify continuous line of object colors
 - Filter out noise pixels
 - Identify colors to form pixel group

imming

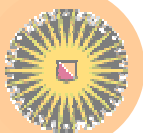


ht
nl

Bayesian Filter

- Why should you care?
 - Robot and environmental state estimation is a fundamental problem!
- Nearly all algorithms that exist for spatial reasoning make use of this approach
 - If you're working in mobile robotics, you'll see it over and over!
 - Very important to understand and appreciate
- Efficient state estimator
 - Recursively compute the robot's current state based on the previous state of the robot

■ *What is the robot's state?*



Perception: Localization with Uncertainty

Initial state
detects nothing:



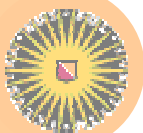
Moves and
detects landmark:

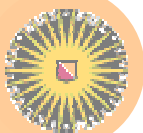


Moves and
detects nothing:



Moves and
detects landmark:

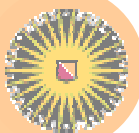
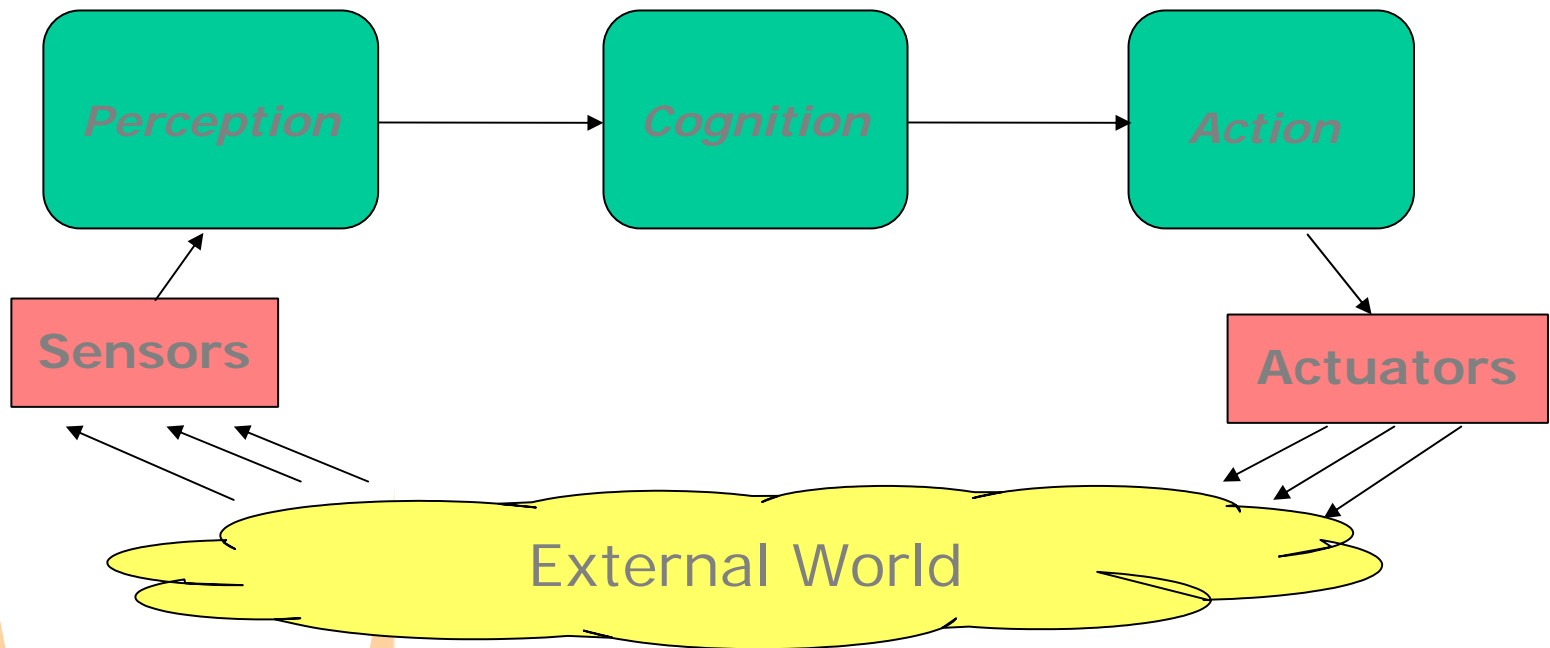




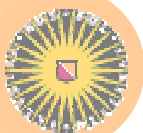
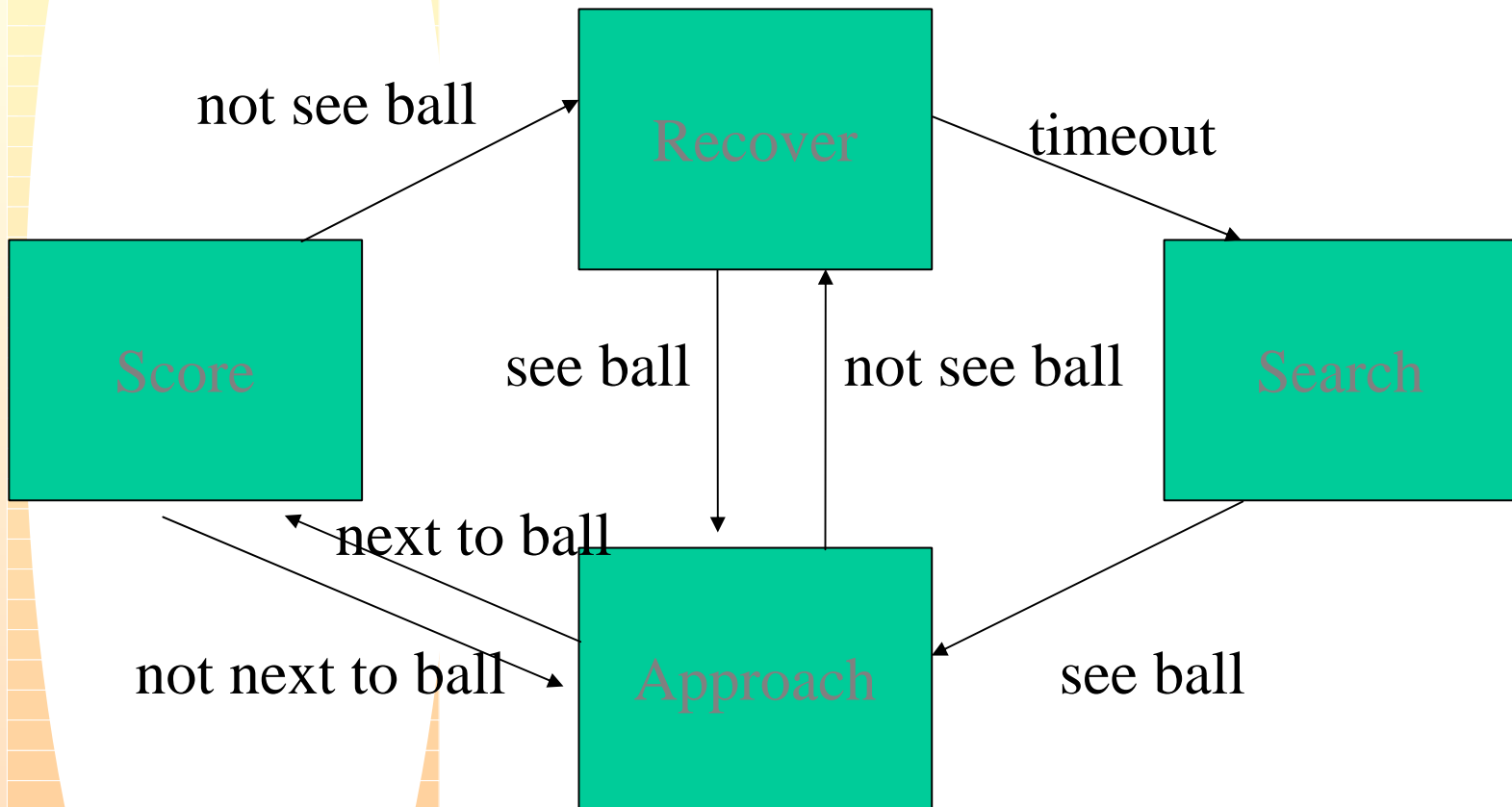
Planning and Motion

- **Task Planning**
- **Motion Planning and Navigation**
- **Mapping**
- **Motion Planning with Uncertainty
(Probabilistic Robotics)**

Intelligent Complete Robot

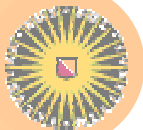


Task Planning: Behavior selection



Action: Motion

- Four-legged walking (several joints with degrees of liberty)
- Head motion (2 joints, 3 degrees of liberty)
- How to generate complex behaviors (turning, kicking?)
- Kinematics: relation between the control inputs and the robot motion
 - Forward kinematics problem
 - Given the control inputs, how does the robot move
 - Inverse kinematics problem
 - Given a desired motion, which control inputs to choose



Robot Motion

- A 51-parameter structure is used to specify the gait of the robot.

Leg Parameters:

Neutral Kinematic Position (3x4)

Lifting Velocity (3x4)

Lift Time (1x4)

Set Down Velocity (3x4)

Set Down Time (1x4)

Global Parameters:

Height of Body (1)

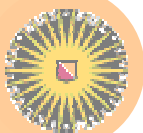
Angle of Body (1)

Hop Amplitude (1)

Sway Amplitude (1)

Walk Period (1)

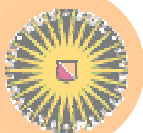
Height of Legs (2)





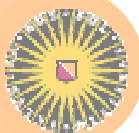
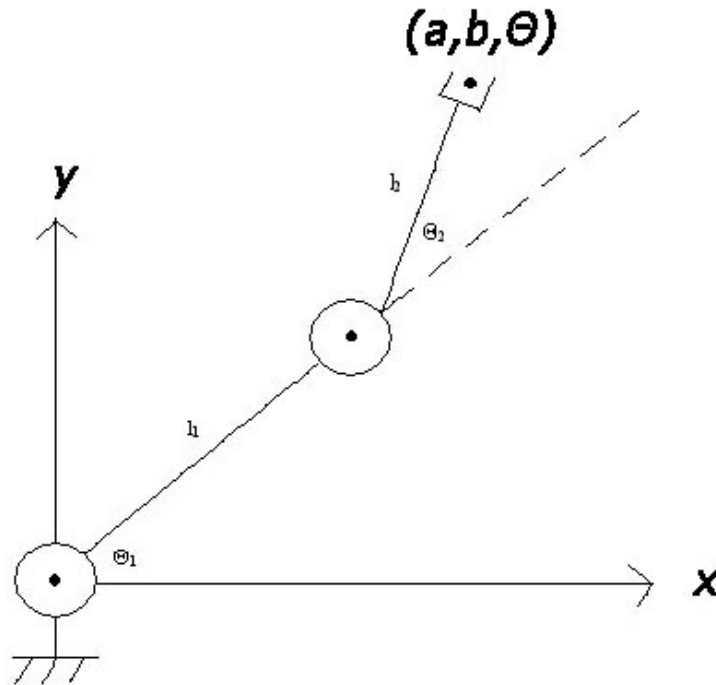
Approaches for Parameter Setting

- Trial and error
 - Tedious, but controlled, and provides knowledge of parameters
- Search
 - Large parameter space, local vs. global optima
- Adaptation
 - Controlled change by feedback



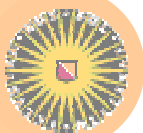
Forward Kinematics

- Determines position in space based on joint configuration



Inverse Kinematics

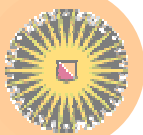
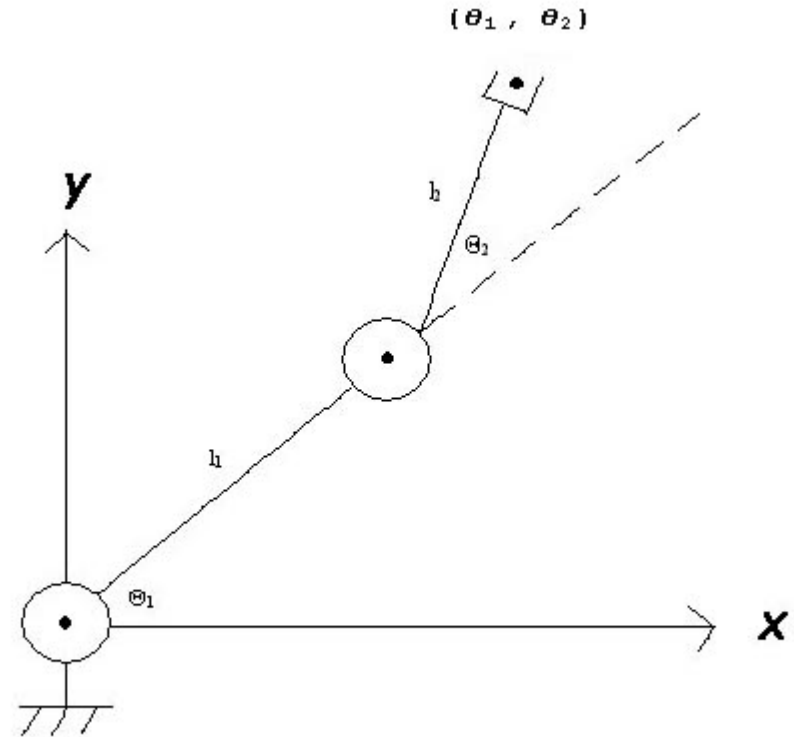
- Compute joint configuration to attain a desired position and orientation of end effector (tool) of robot
- More complex than forward kinematics
- Often also involves path-planning to attain joint configuration from the current one
- Usually solved algebraically or geometrically, but needs model of the robot!
- Possibly no solution, one solution, or multiple solutions



An example

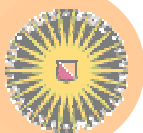
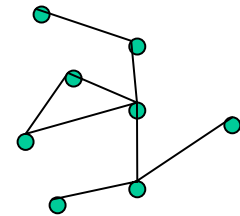
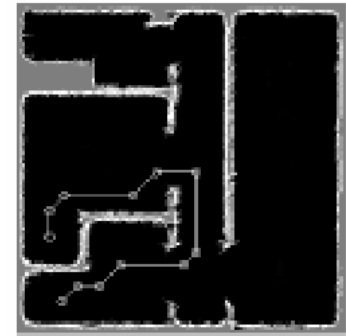
Let's assume $l_1 = l_2$

What is the configuration of the joint-angles if the end-effector is located at (l_1, l_2) ?



World Models (I)

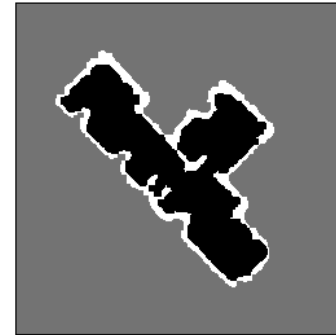
- Representations of the environment are usually built by means of:
 - **Metric maps**: explicitly reproduce the metrical structure of the domain
 - good for location, hard for planning
 - e.g., *Evidence grids*
 - **Topological maps**: represent the environment as a set of meaningful regions.
- In our architecture we use *both* representations
 - good for planning, hard for location



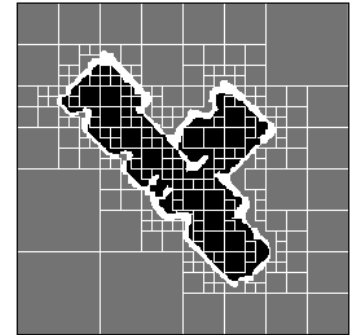
World Models (II)

Topological Map Extraction

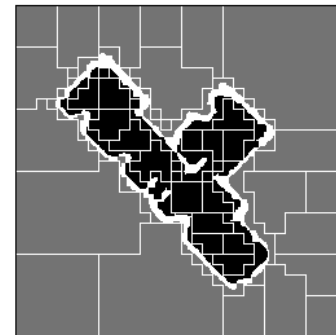
- (a) **Metric map thresholding**
 - cell occupancy values
- (b) **Hierarchical split**
 - pyramidal cell structure
- (c) **Interlevel merging**
 - homogeneous cells fusion
- (d) **Intralevel merging**
 - homogeneous cell classification



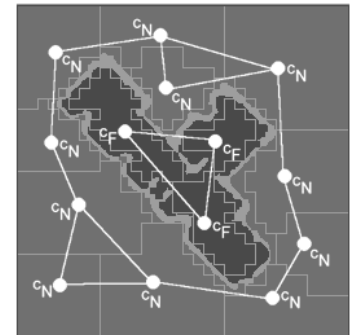
(a)



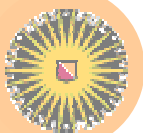
(b)



(c)

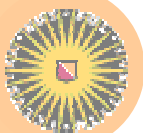
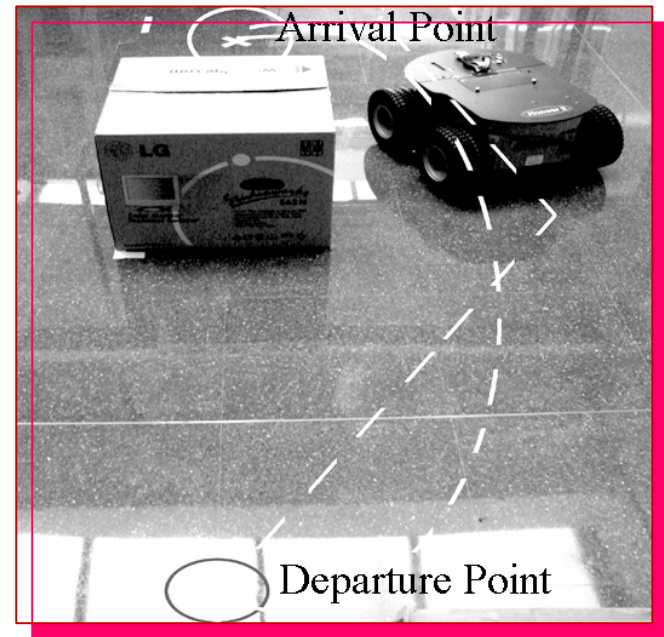


(d)



Navigation (I)

- **Navigation** consists of finding and tracking a safe path from a departure point to a goal.
- Navigation **architectures** belong to three broad categories: deliberative, reactive and hybrid.



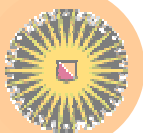
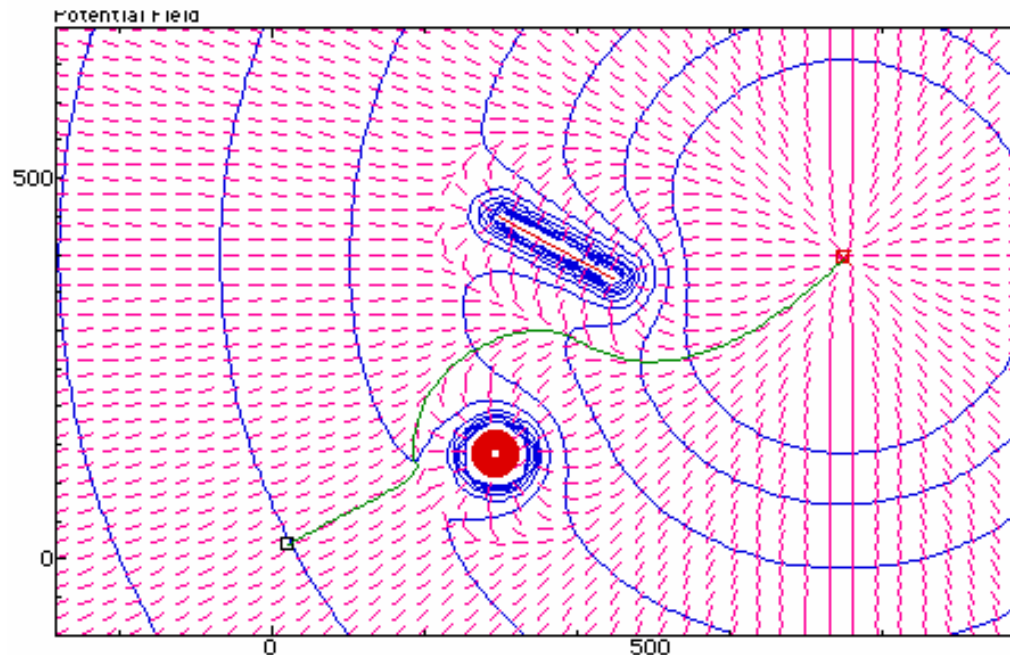
Navigation (II)

- **Deliberative schemes** require extensive world knowledge to build high-level plans
 - Usually they use the *sense-model-plan-act* cycle
 - **problem 1**: inability to react rapidly
 - **problem 2**: not suitable for (partially) unknown environments.
- **Reactive schemes** try to couple *sensors* and *actuators* to achieve a *fast* response.
 - Easily combine several sensors and goals,
 - **problem 1**: the emergent behaviour may be *unpredictable*
 - **problem 2**: the emergent behaviour may be *inefficient* (prone to fall in local traps).
- **Hybrid schemas** get the best of both approaches.

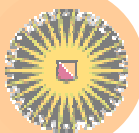
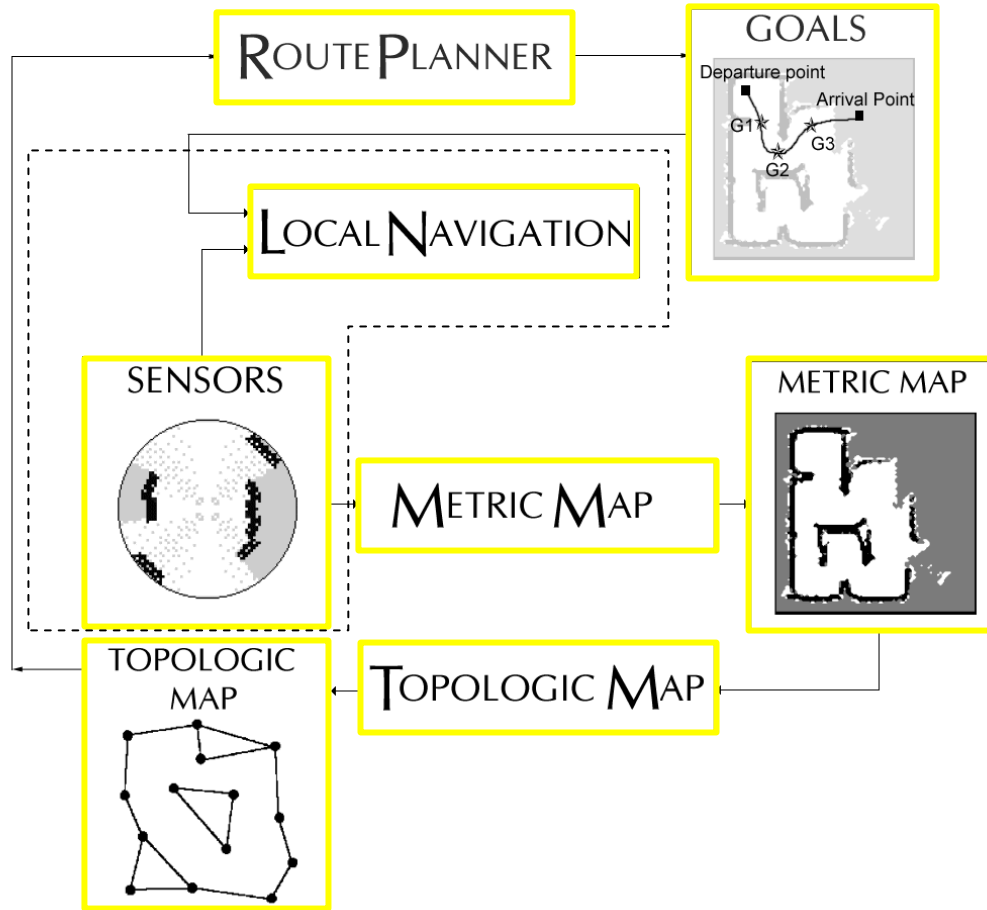


Reactive system using Potential Fields

- Create repulsion force around obstacles plus attraction force to the goal



The hybrid architecture



Discussion

- Robot Controllers need:
 - 1) Perception of the environment; classifying and measuring distances to objects
 - 2) Self localization (dealing with uncertainty)
 - 3) Obstacle-free navigation; path-planning, reactive behaviour
 - 4) Task planning; what is the current/next goal?
- For multiple-robots the robots also need to:
 - 1) Communicate (e.g. shared world models)
 - 2) Negotiate (goal division among team members)

