Robotics/Perception II

Mariusz Wzorek

Artificial Intelligence and Integrated Computer Systems Division (AIICS)

Outline

- Sensors - summary
- Computer systems
- Robotic architectures
- Mapping and Localization
- Motion planning
- Motion control

AIICS Autonomous UAV Technologies

DARPA Challenges - application examples/challenges

2004 Grand Challenge - 240 km route in the Mojave Desert.

None of the vehicles finished the route. Sandstorm (Red Team,CMU) - 11.78 km of the course before getting hung up on a rock.

2005 Grand Challenge - Three narrow tunnels and more than 100 sharp left and right turns.

Vehicle	Team Name	Team Home	Time Taken (h:m)	Result
Stanley	Stanford Racing Team	Stanford University, Palo Alto, California	6:54	First place
Sandstorm	Red Team	Carnegie Mellon University, Pittsburgh, Pennsylvania	7:05	Second place
H1ghlander	Red Team	Carnegie Mellon University, Pittsburgh, Pennsylvania	7:14	Third place
Kat-5	Team Gray	The Gray Insurance Company, Metairie, Louisiana	7:30	Fourth place
TerraMax	Team TerraMax	Oshkosh Truck Corporation, Oshkosh, Wisconsin	12:51	Over 10 hour limit, fifth place

2007 Urban Challenge - 96 km urban area course, to be completed in less than 6 hours. Rules included obeying all traffic regulations while negotiating with other traffic and obstacles and merging into traffic.

2012 Robotics Challenge - ongoing focusing on humanoid robots.

AIICS Autonomous UAV Technologies

Anatomy of Robots

Robots consist of:

- Motion mechanism
  - Wheels, belts, legs, propellers, rotors
- Manipulators
  - Arms, grippers
- Sensors
- Computer systems
  - Microcontrollers, embedded systems
- Body/Frame

Mariusz Wzorek
Artificial Intelligence & Integrated Computer Systems Division
Department of Computer and Information Science
Linköping University, Sweden

Sensors

Summary of most commonly used sensor for mobile robots.

	Any weather	Any light	Detection in 15m	Fast response	Weight	Affordable
CCD Camera/stereo/ Omnidirectional/o. flow			👍	👍	👍	👍
Ultrasonic		👍		👍	👍	👍
Scanning laser		👍	👍	👍		👍
3D Scanning laser		👍	👍			👍*
Millimeter Wave Radar	👍	👍	👍	👍		

* scanning laser on a tilting unit

Mariusz Wzorek
Artificial Intelligence & Integrated Computer Systems Division
Department of Computer and Information Science
Linköping University, Sweden

AIICS Autonomous UAV Technologies

Sensors - common interfaces

- analog - voltage level
- digital:
  - serial connections e.g.:
    - RS232
    - I2C
    - SPI
    - USB
  - pulse width modulation - PWM
  - …

Mariusz Wzorek
Artificial Intelligence & Integrated Computer Systems Division
Department of Computer and Information Science
Linköping University, Sweden

AIICS Autonomous UAV Technologies

Computer systems

Many challenges and trade-offs!

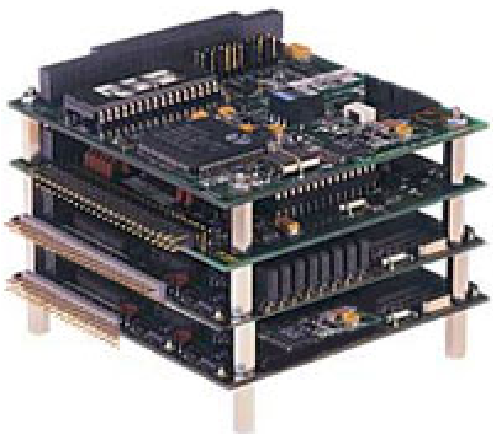
- power consumption
- size & weight
- computational power
- robustness
- different operational conditions - moisture, temperature, dirt, vibrations, etc.

AIICS Autonomous UAV Technologies

Computer systems cont'd

PC104 - standardised form factor

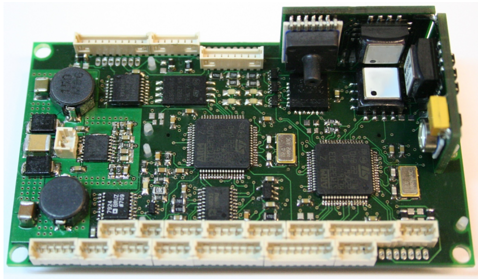
- industrial grade,
- relatively small size,
- highly configurable,
- >200 vendors world-wide,
- variety of components
- variety of processors

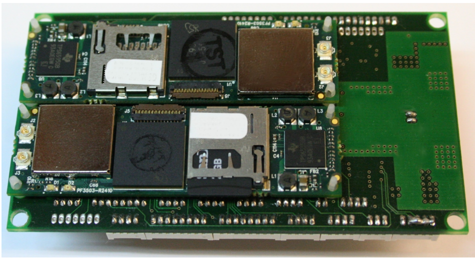


Computer systems cont'd

Custom made embedded systems

- with integrated sensor suite
- micro scale, light weight
- fitted for platform design

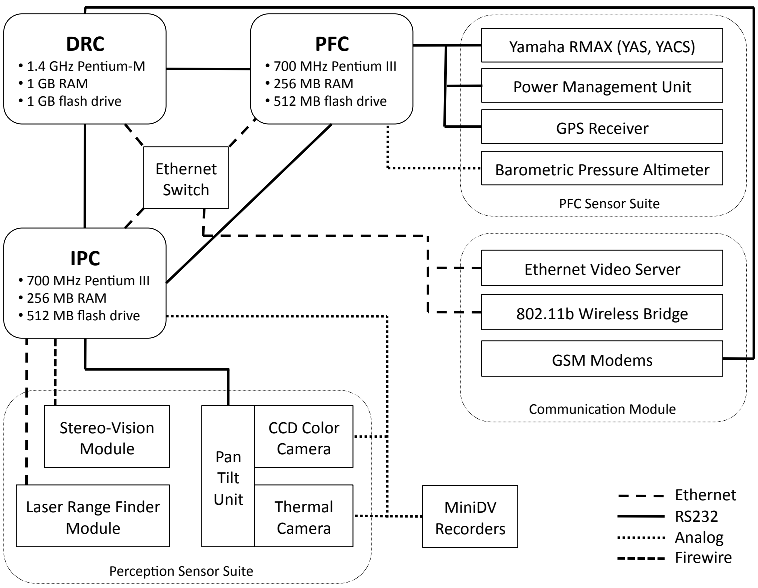




- application specific

Computer systems cont'd

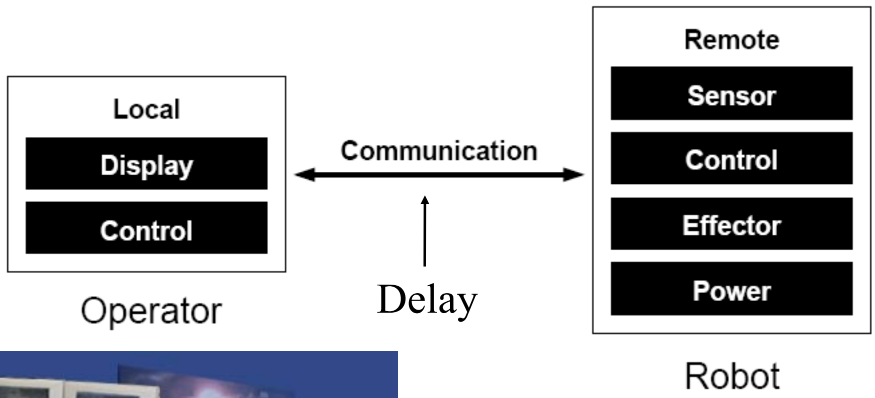
Example system design:

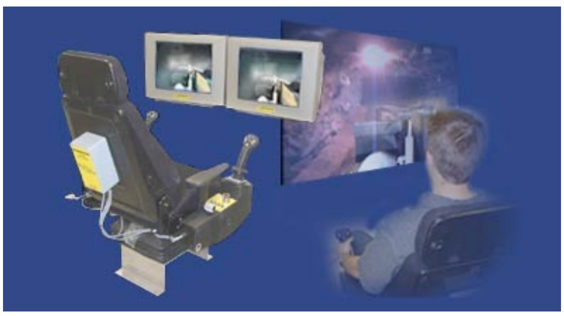


Outline

- Sensors - summary
- Computer systems
- Robotic architectures
- Mapping and Localization
- Motion planning
- Motion control

Telesystems/Telemanipulators





Video

Telepresence

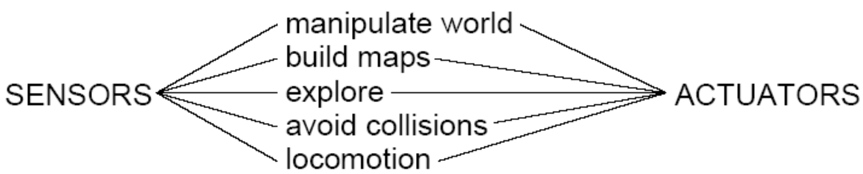
Experience of being fully present at live event without actually being there
- see the environment through robot’s cameras
- feel the surrounding through robot sensors

Realized by for example:
- stereo vision
- sound feedback
- cameras that follow the operator’s head movements
- force feedback and tactile sensing
- VR system

Reactive systems

SENSE ⟺ ACT

Robot responses **directly to sensor stimuli**.
Intelligence emerges from combination of simple behaviours.
E.g. - subsumption architecture (Rodney Brooks, MIT):
- concurrent behaviours
- higher levels subsume behaviours of lower layers
- no internal representation
- finite state machines

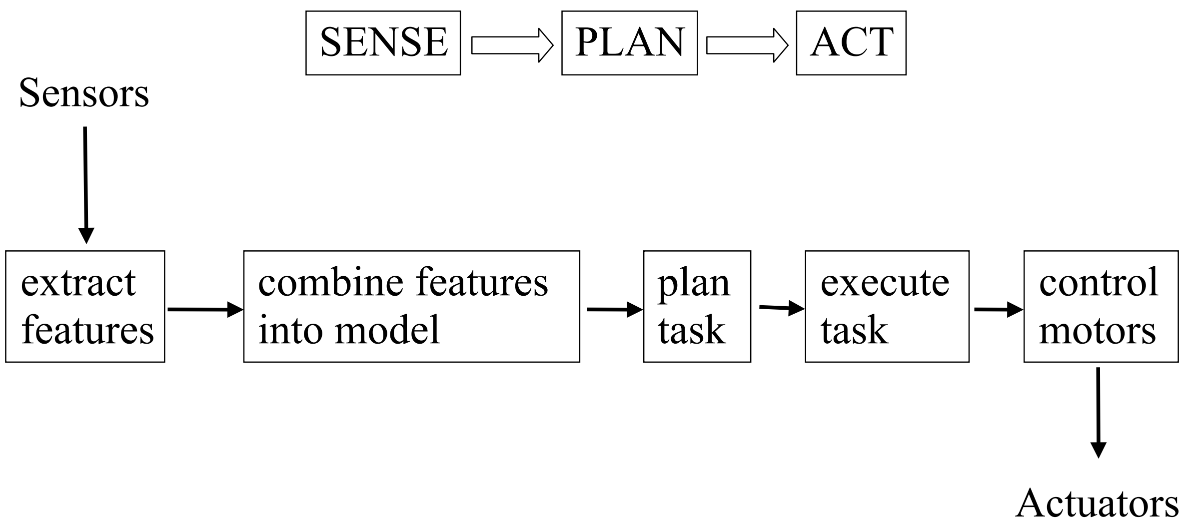


Video

Limitations of Reactive Systems

- Agents without environment models must have sufficient information available from local environment
- If decisions are based on *local* environment, how does it take into account *non-local* information (i.e., it has a "short-term" view)
- Difficult to make reactive agents that learn
- Since behaviour emerges from component interactions plus environment, it is hard to see how to *engineer* specific agents (no principled methodology exists)
- It is hard to engineer agents with large numbers of behaviours (dynamics of interactions become too complex to understand)

Hierarchical Systems



Hierarchical Systems (Shakey Example)

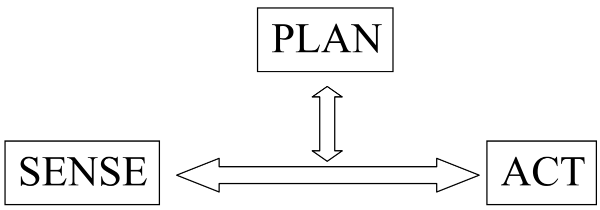
Shakey (1966 - 1972)

- Developed at the Stanford Research Institute
- Used STRIPS planner (operators, pre and post conditions)
- Navigated in an office environment, trying to satisfy a goal given to it on a teletype. It would, depending on the goal and circumstances, navigate around obstacles consisting of large painted blocks and wedges, push them out of the way, or push them to some desired location.
- Primary sensor: black-and-white television camera
- Sting symbolic logic model of the world in the form of first order predicate calculus
- Very careful engineering of the environment

Video

AIICS Autonomous UAV Technologies

Hybrid systems (three-layer architecture)



Deliberative layer (planning)

Executive layer (sequencing)

Reactive layer (skills, control)

Mariusz Wzorek
Artificial Intelligence & Integrated Computer Systems Division
Department of Computer and Information Science
Linköping University, Sweden

AIICS Autonomous UAV Technologies

Hybrid systems (Minerva Example)

Tour guide at the Smithsonian's National Museum of American History (1998)

high-level control and learning (mission planning, scheduling)
human interaction modules ("emotional" FSA, Web interface)
navigation modules (localization, map learning, path planning)
hardware interface modules (motors, sensors, Internet)

Table 1: Minerva's layered software architecture

Video

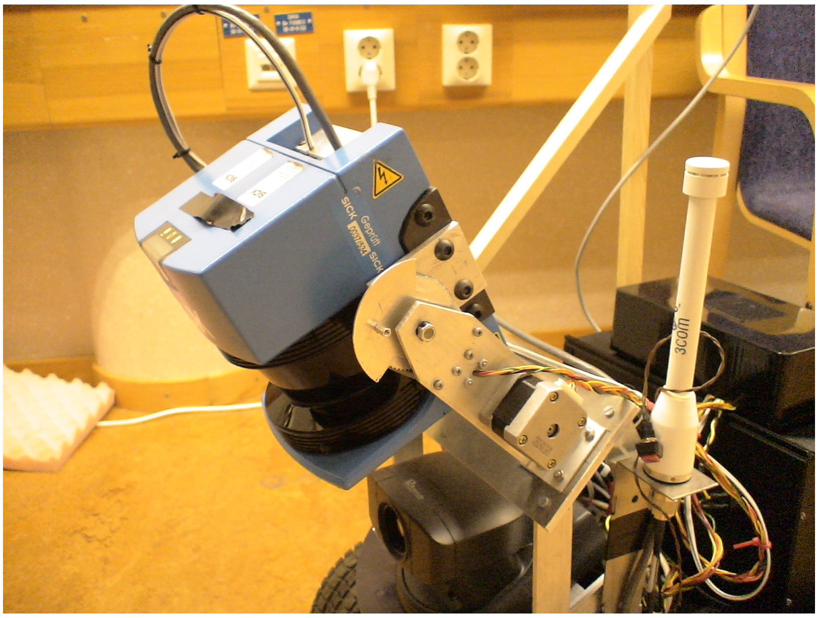
AIICS Autonomous UAV Technologies

Outline

- Sensors - summary
- Computer systems
- Robotic architectures
- Mapping and Localization
- Motion planning
- Motion control

Autonomous UAV Technologies

Laser Range Finder

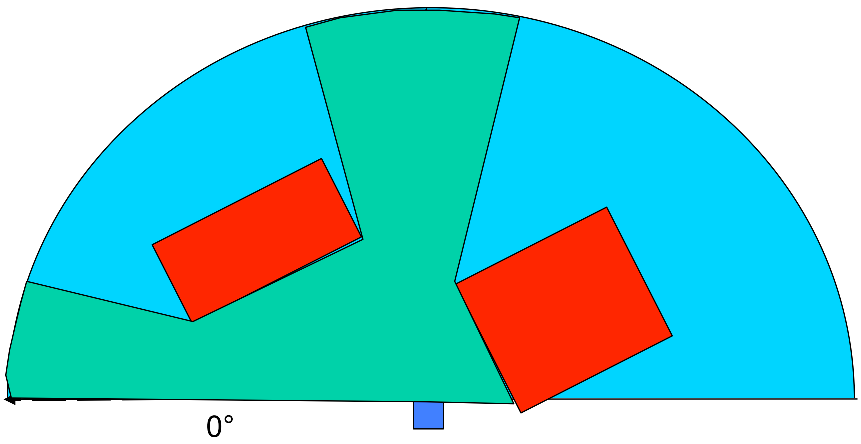


Mariusz Wzorek

Artificial Intelligence & Integrated Computer Systems Division
Department of Computer and Information Science
Linköping University, Sweden

Autonomous UAV Technologies

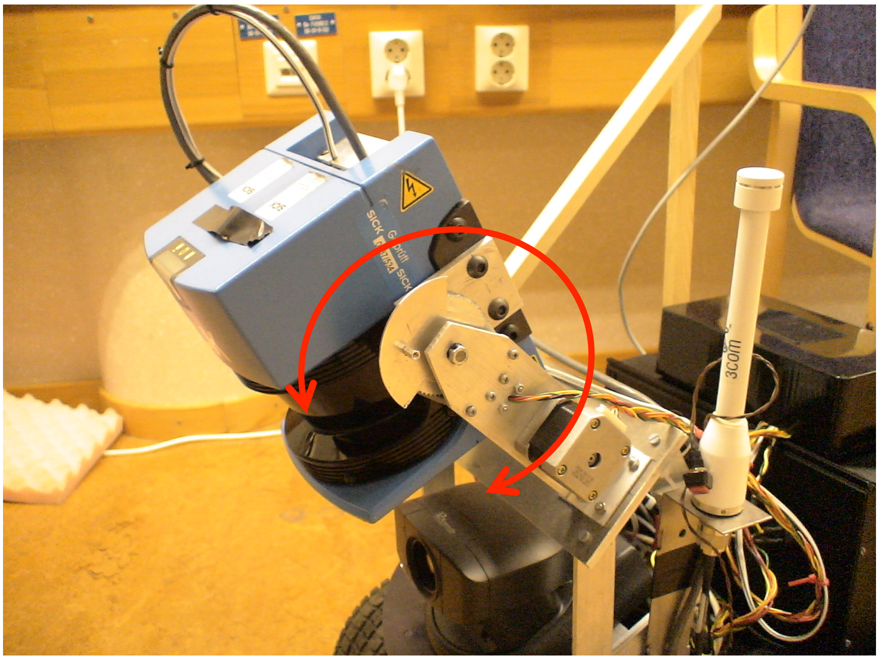
Laser Range Finder - the principle



View from top

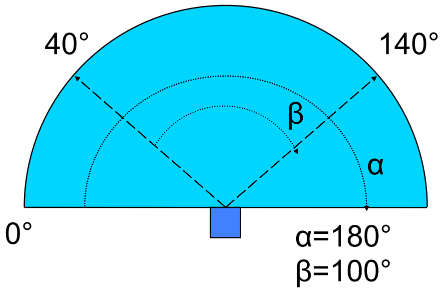
Video

Laser Range Finder - the principle cont'd



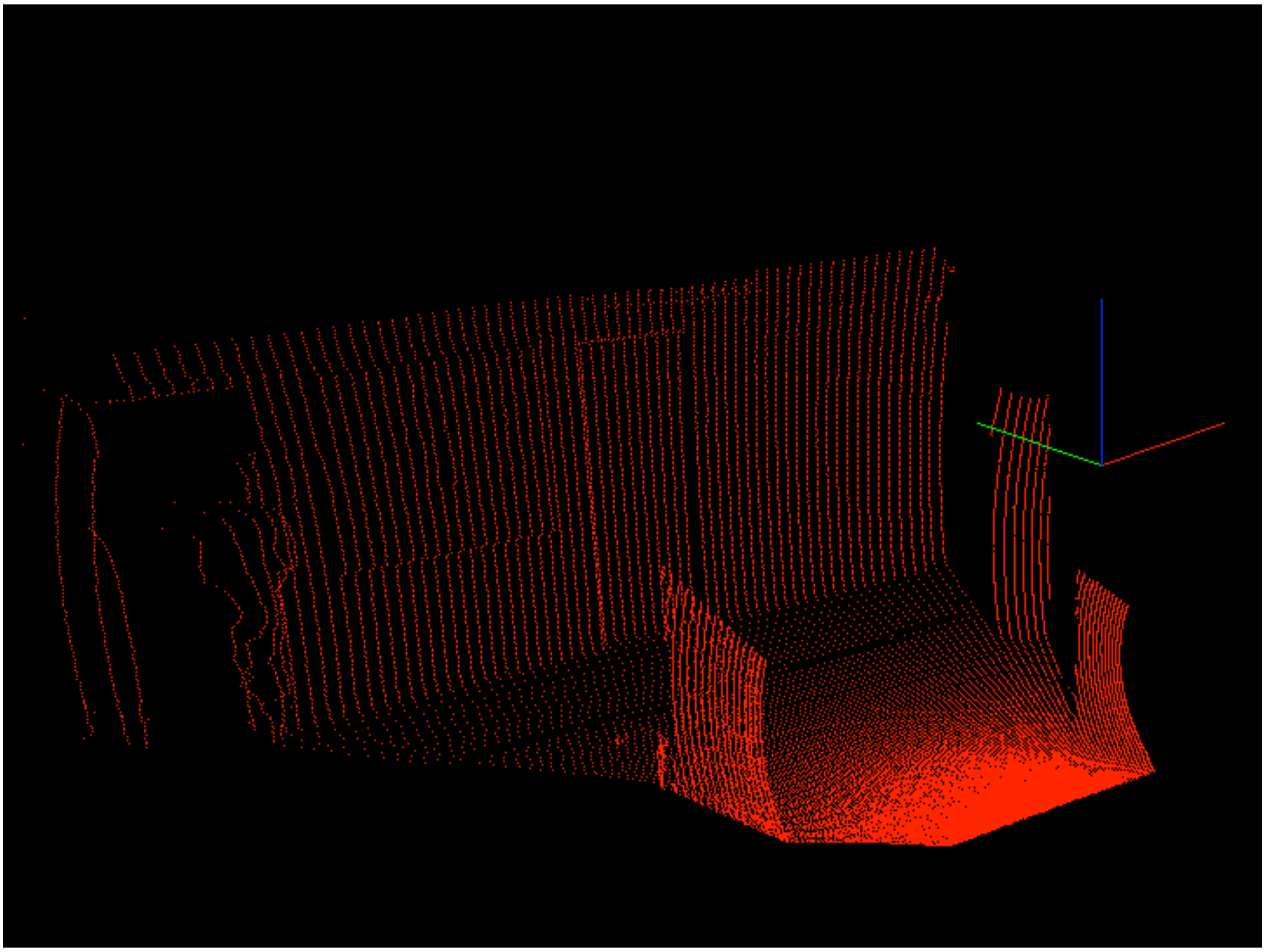
SICK LMS Resolution Modes

Angular range	Angular resolution
0°..100°	1°
0°..100°	0.5°
0°..100°	0.25
0°..180°	1°
0°..180°	0.5°



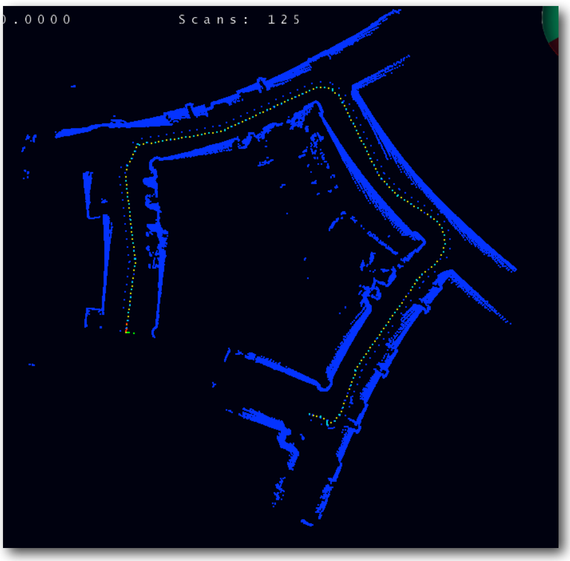
View from top

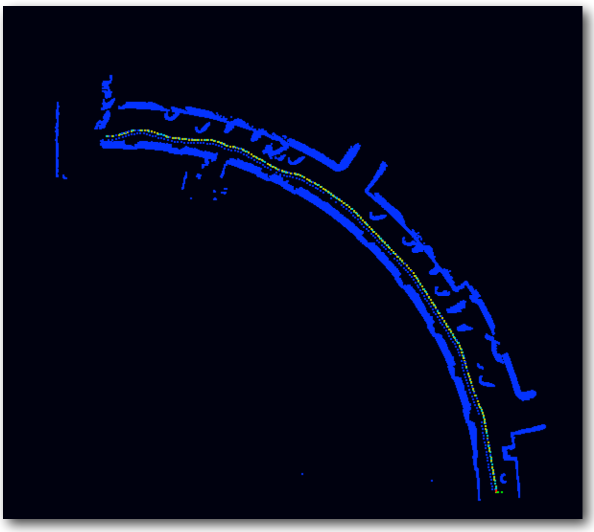
Mode	Measurement Range
mm mode	0..8191mm = 8.191meters
cm mode	0..8191cm = 81.91meters



AIICS Autonomous UAV Technologies

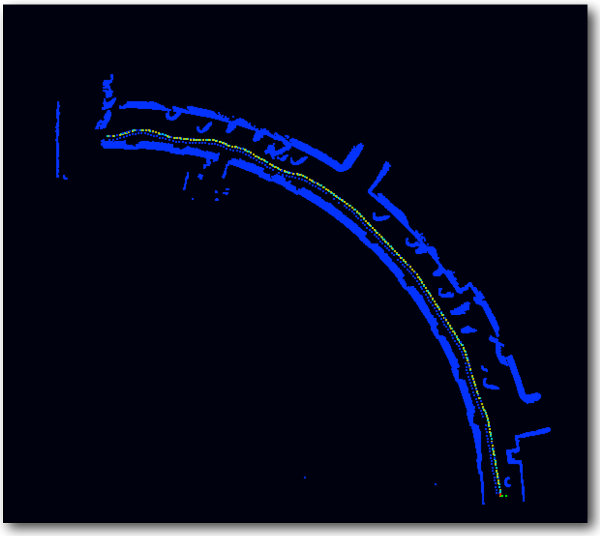
Mapping





Based only on odometry + range data

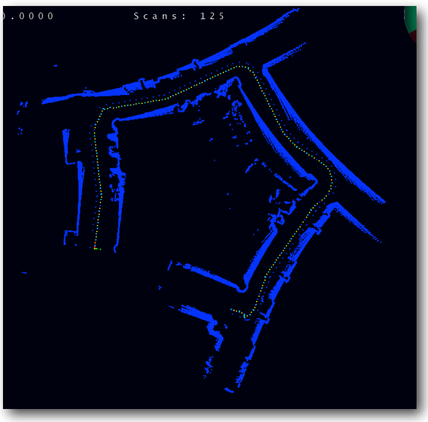
Mapping cont’d

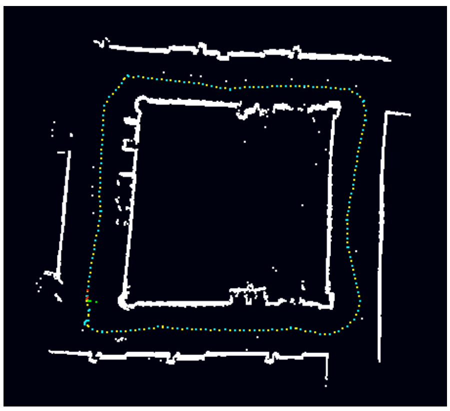


Based only on odometry + range data + matching

3D Mapping video

Mapping cont’d





Based only on odometry + range data + matching

Robotic perception

Robot perception viewed as **temporal inference from sequences of actions and measurements** → dynamic Bayes network of first order Markov process

Example: **localization** - holonomic robot with range sensor, estimate pose while moving

$\vec{x}_t = (x_t, y_t, \theta)^T$ not observable current state

$\vec{a}_t = (v_t, \omega_t)^T$ known action

$\vec{o}_t = (range1, range2, \cdots)^T$ observable sensor reading

$\mathrm{Bel}(\vec{x}_t) = \mathrm{p}(\vec{x}_t | \vec{o}_{1:t}, \vec{a}_{1:t-1})$ current belief state (captures past)

next belief state? → Bayesian inference problem

Recursive filtering equation

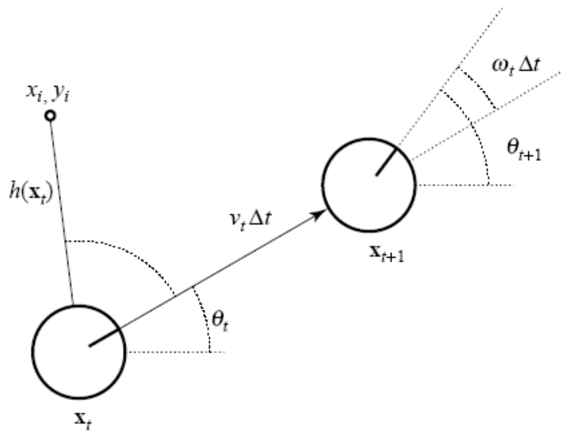
$$\mathrm{Bel}(\vec{x}_t) = \alpha \underbrace{\mathrm{p}(\vec{o}_t | \vec{x}_t)}_{\text{sensor model}} \int \underbrace{\mathrm{p}(\vec{x}_t | \vec{x}_{t-1}, \vec{a}_{t-1})}_{\text{motion model}} \underbrace{\mathrm{Bel}(\vec{x}_{t-1})}_{\text{previous belief state}} d\vec{x}_t$$

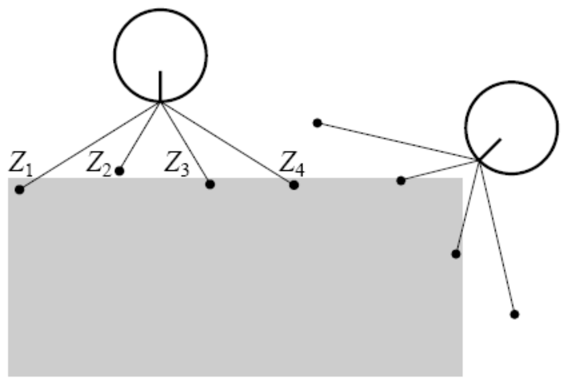
using Bayes' rule, Markov assumption, theorem of total probability

Motion model: deterministic state prediction + noise

Sensor model: likelihood of making observation o_t when robot is in state x_t

Motion and sensor model





Assume Gaussian noise in motion prediction, sensor range measurements

Localization algorithms

Particle filter (Monte Carlo localization):

- belief state - a collection of particles that correspond to states
- belief state sampled, each sample weighted by likelihood it assigns to new evidence, population resampled using weights

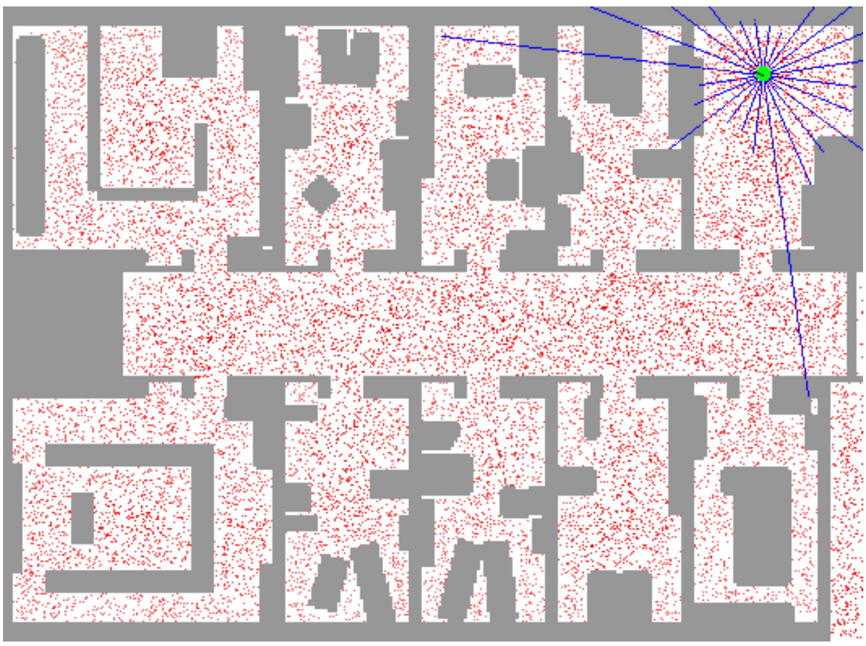
Kalman filter:

- belief state - a single multivariate Gaussian
- each step maps a Gaussian into a new Gaussian, i.e. it computes a new mean and covariance matrix from the previous mean and covariance matrix
- assumes linear motion and measurement models (linearization →extended KF)

Global Localization (Particle Filter Examples)

$$\mathrm{Bel}(\vec{x}_t) = \alpha \underbrace{p(\vec{o}_t|\vec{x}_t)}_{\text{sensor model}} \int \underbrace{p(\vec{x}_t|\vec{x}_{t-1},\vec{a}_{t-1})}_{\text{motion model}} \underbrace{\mathrm{Bel}(\vec{x}_{t-1})}_{\text{previous belief state}} \, d\vec{x}_t$$

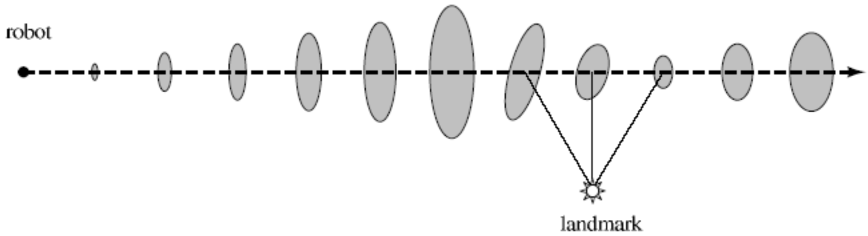
Global Localization (Particle Filter Examples; sonar vs laser)



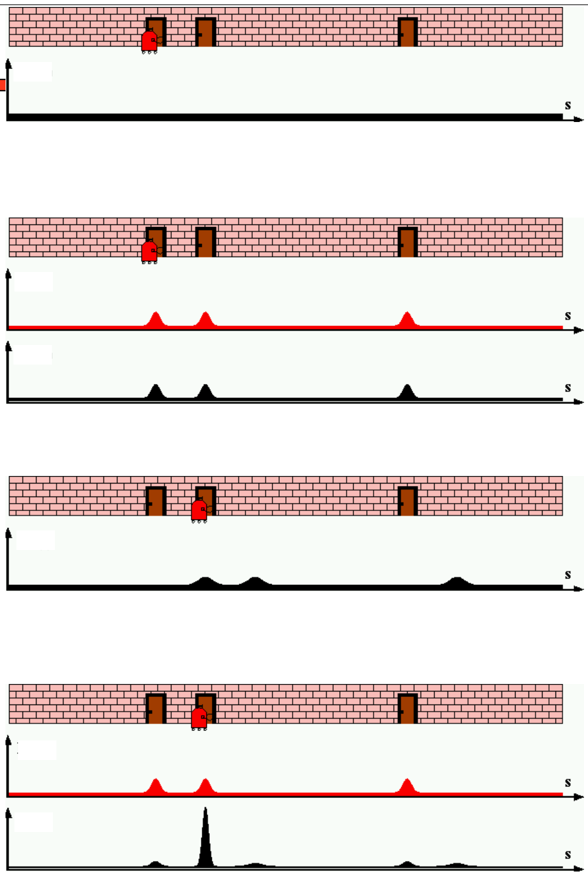
Video_sonar

Video_laser

Extended Kalman Filter (EKF) - example



EKF example



SLAM

Localization: given map and observed landmarks, update pose distribution

Mapping: given pose and observed landmarks, update map distribution

SLAM: given observed landmarks, update pose and map distribution

Probabilistic formulation of SLAM:

add landmark locations $L_1, \ldots, L_k$ to the state vector, proceed as for localization

Problems:

- dimensionality of map features has to be adjusted dynamically
- identification of already mapped features

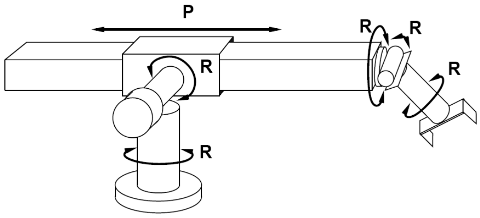
Laser-based SLAM Video Distributed SLAM Video

Outline

- Sensors - summary
- Computer systems
- Robotic architectures
- Mapping and Localization
- Motion planning
- Motion control

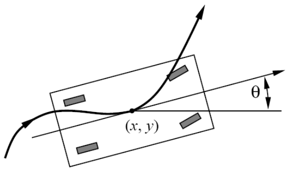
Degree of freedom (DOF)

One DOF for each independent direction in which robot can move



Configuration of robot specified by 6 numbers => 6 DOFs

Non-holonomic: number of effective DOFs less than controllable DOFs



Car has 3 effective DOF and 2 controls =>non-holonomic

Motion Planning

Motion types:

- point-to-point
- compliant motion (screwing, pushing boxes)

Representations: configuration space vs workspace

Kinematic state: robot's configuration (location, orientation, joint angles), no velocities, no forces

Path planning: find path from one configuration to another

Problem: continuous state space, can be high-dimensional

AIICS Autonomous UAV Technologies

Motion Planning - representations

Workspace - physical 3D space (e.g. joint positions)

Robot has rigid body of finite size

Well-suited for collision checking

Problem: linkage constraints (not all workspace coordinates attainable) makes path planning difficult in workspace

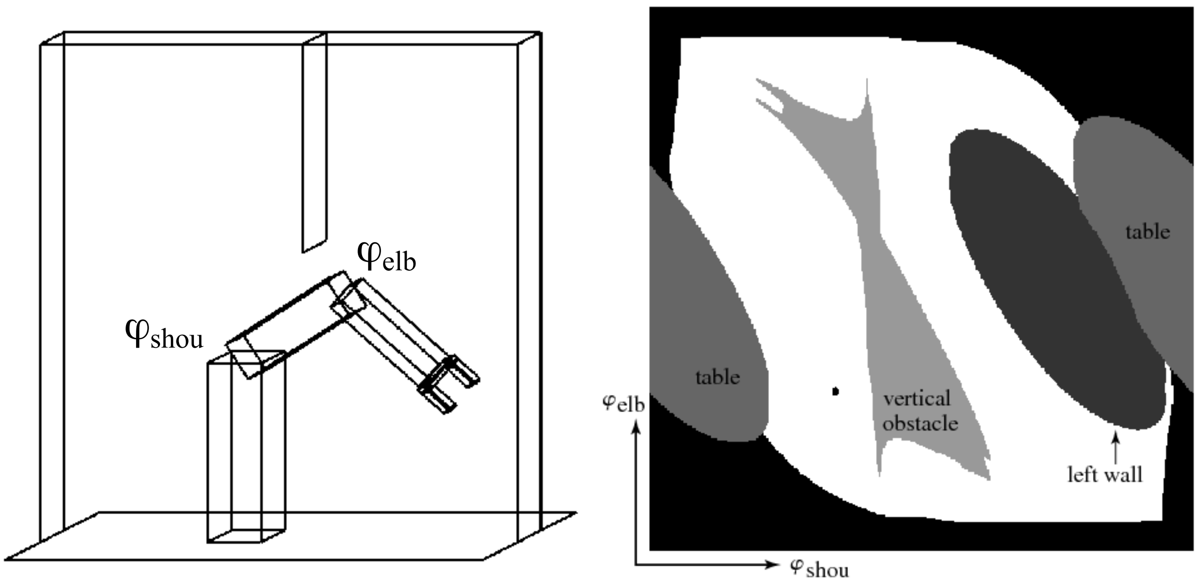
Configuration Space - space of robot states (e.g. joint angles)

Robot is a point, obstacles have complex shapes

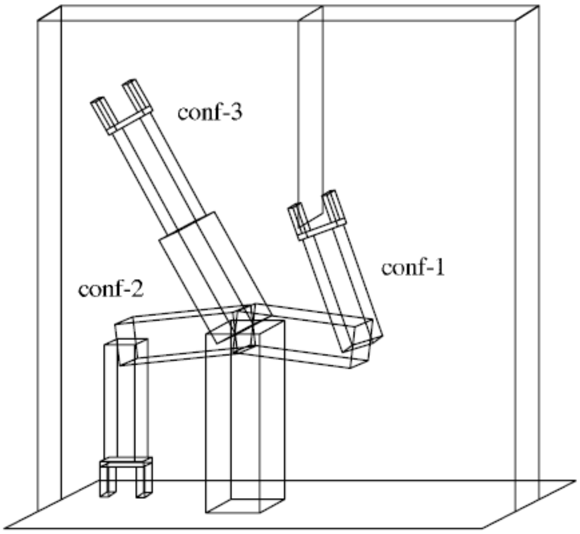
Problem: tasks are expressed in workspace coordinates, obstacle representation problematic

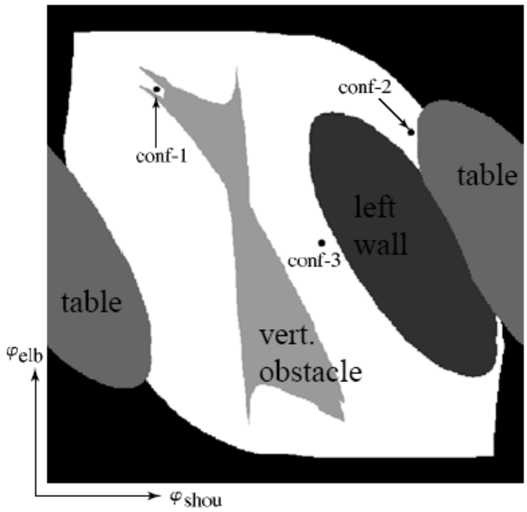
AIICS Autonomous UAV Technologies

Workspace vs. Configuration Space



Workspace vs. Configuration Space





Motion Planning - representations

- Free space (attainable configurations) ->occupied space (not attainable configurations, obstacles)
- Planner may generate configuration in configuration space, apply inverse kinematics and check in workspace for obstacles

Inverse kinematics (often hard and ill-posed problem)

Workspace → Configuration Space

Configuration Space → Workspace

Kinematics (simple, well-posed problem)

Path Planning

Basic problem: convert infinite number of states into finite state space

Cell decomposition:

- divide up space into simple *cells*,
- each of which can be traversed “easily”

Skeletonization:

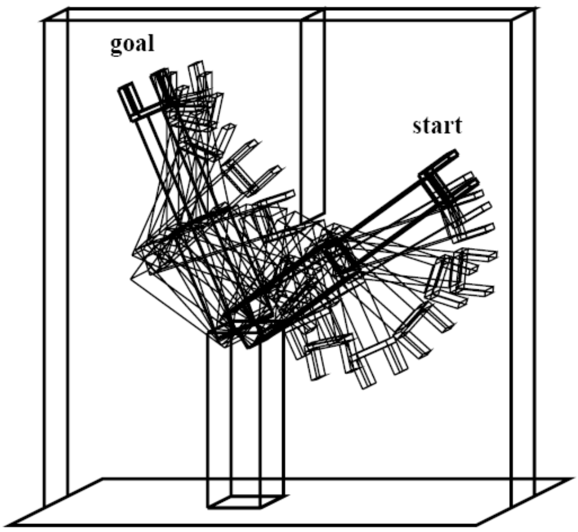
- identify a finite number of easily connected points/lines
- form a graph such that any two points are connected by a path

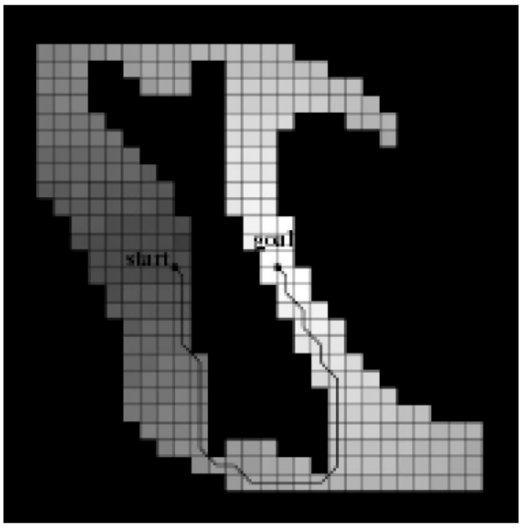
Graph search and colouring algorithms

Assumptions: motion deterministic, localization exact, static scenes

Not robust with respect to small motion errors, does not consider limits due to robot dynamics

Cell Decomposition





Grayscale shading - cost from the grid cell do the goal

Cell Decomposition

Problem: may be no path in pure free space cells

Soundness

(wrong solution if cells are mixed)

vs.

Completeness

(no solution if only pure free cells considered)

Solution: recursive decomposition of mixed (free+obstacle) cells or exact decomposition

Doesn't scale well for higher dimensions

Skeletonization

Visibility graphs: find lines connecting obstacle vertices through free space, build and search graph; not for higher dimensions

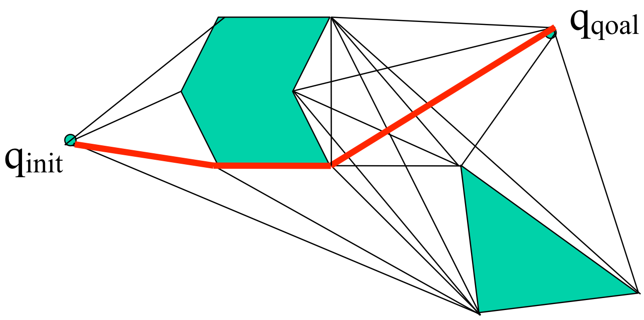
Voronoi graphs: find all points in free space equidistant to two or more obstacles, build and search graph; maximizes clearance, creates unnecessarily large detours, does not scale well for higher dimensions

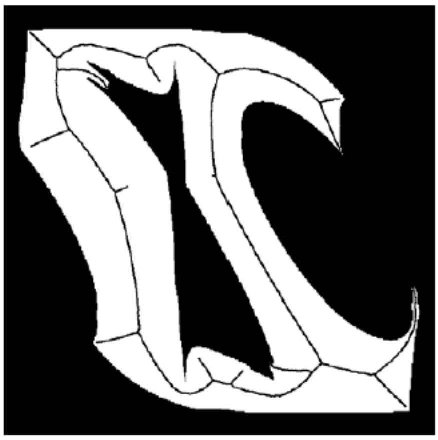
Probabilistic roadmaps:

- generate randomly large number of configurations in free space, build graph (construction phase)
- search graph (query phase)

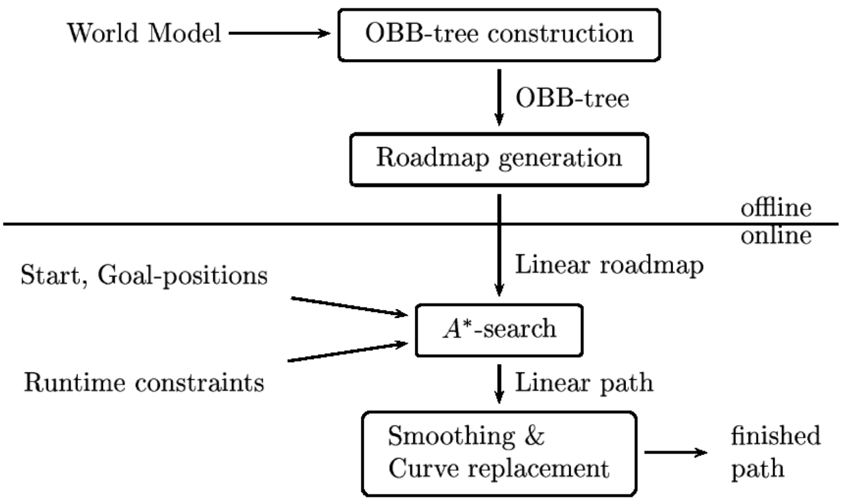
scales better to higher dimensions but incomplete

Visibility and Voronoi Graph

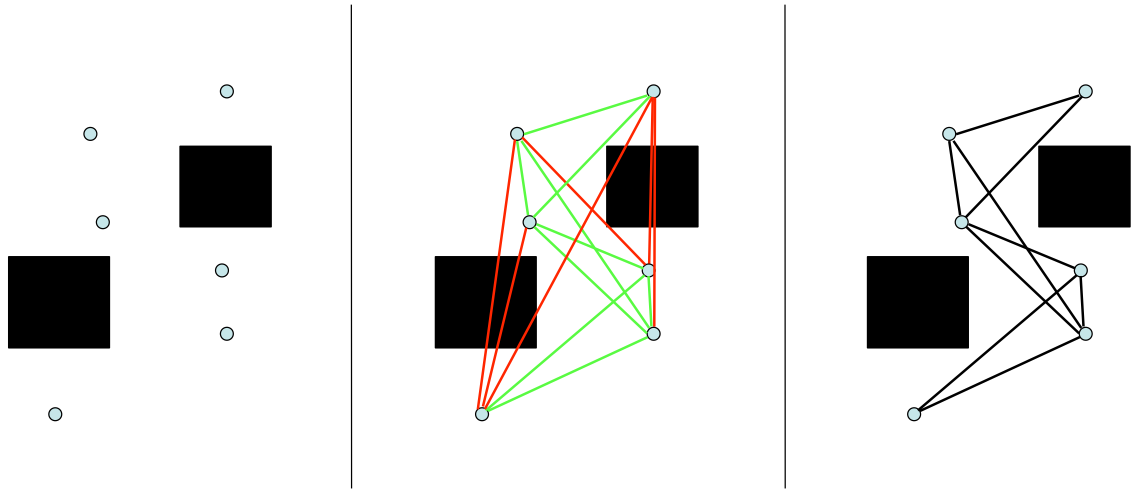




Probabilistic Roadmaps (PRM planning procedure example)



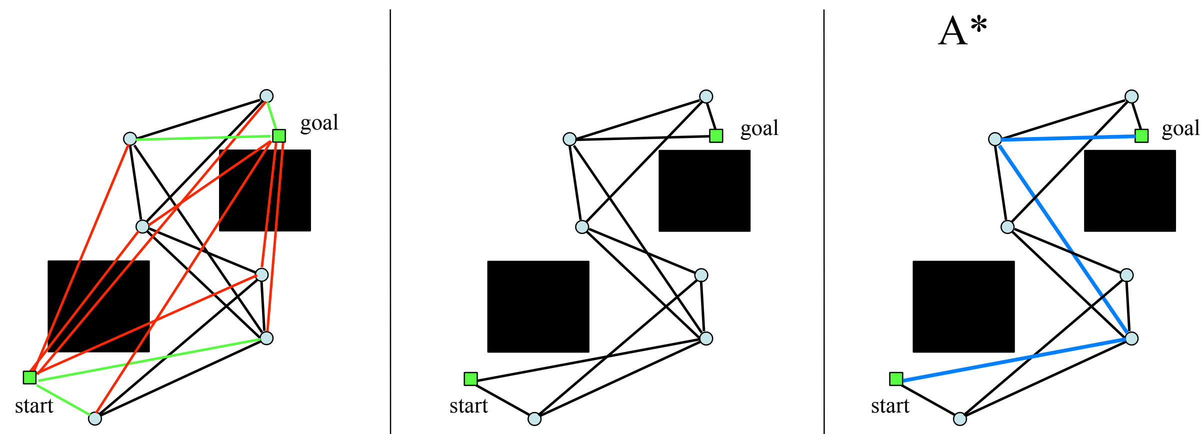
Probabilistic Roadmaps
(PRM planning procedure example: Construction phase)



Generate random configurations

Make connections

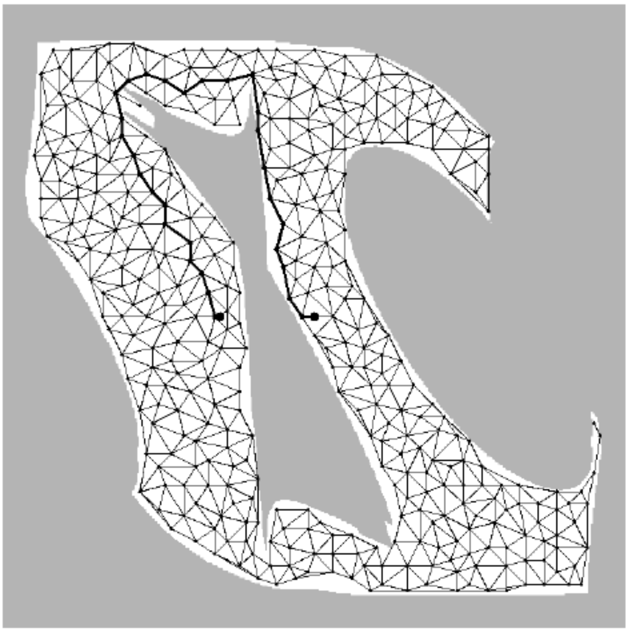
Probabilistic Roadmaps
(PRM planning procedure example: Query Phase)



Add start and goal configurations to the roadmap

Curve Replacement & Smoothing

Probabilistic Roadmaps



Potential Fields

Additional cost function in connectivity graph:
potential grows when closer to obstacles
Maximises clearance from obstacles while minimising path length:

- creates safer paths

Problem: choosing appropriate weights

Motion Control

- Path following involves forces: friction, gravity, inertia,
- Dynamic state: kinematic state + robot's velocities
- Transition models expressed as differential equations
- Path planner assumes robot can follow any path
- Robot's inertia limits manoeuvrability

Problem: including dynamic state in planners makes motion planning intractable

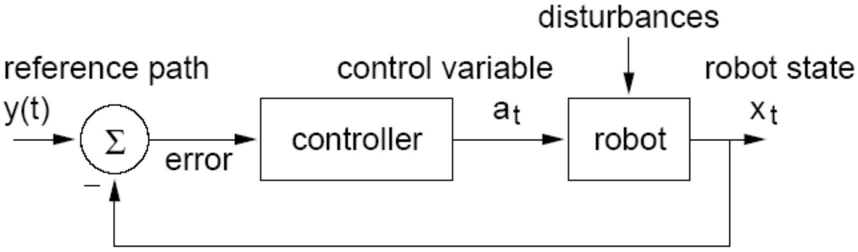
Solution: simple kinematic planners + low-level controller for force calculation

Other solution: motion control without planning: potential field and reactive control

Controllers

- Techniques for **generating robot controls in real time** using feedback from the environment to achieve a control objective
- Reference controller: keep robot on pre-planned path (reference path)
- Optimal controller: controller that optimizes a global cost function, e.g. optimal policies for MDPs
- Stable: small perturbations lead to bounded error between robot and reference signal
- Strictly stable: able to return to reference signal

Closed-loop control

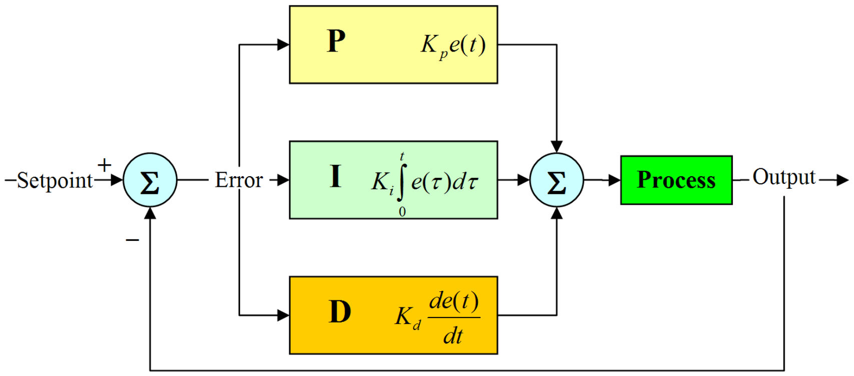


Performance of controller:

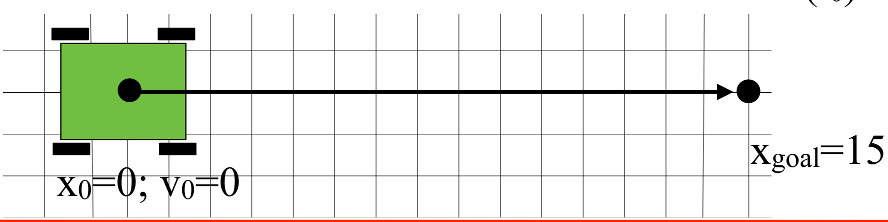
- stability
- overshoot ← inertia
  - steady-state error ← friction
- rise time
  - settling time

P controller: $a_t = K_P(y(t) - x_t)$ K_P gain parameter

PID Example



$e(t_0) = x_{goal} - x_0 = 15$



Closed-loop control

PD controller: $$a_t = K_P(y(t) - x_t) + K_D \frac{\partial(y(t) - x_t)}{\partial t}$$

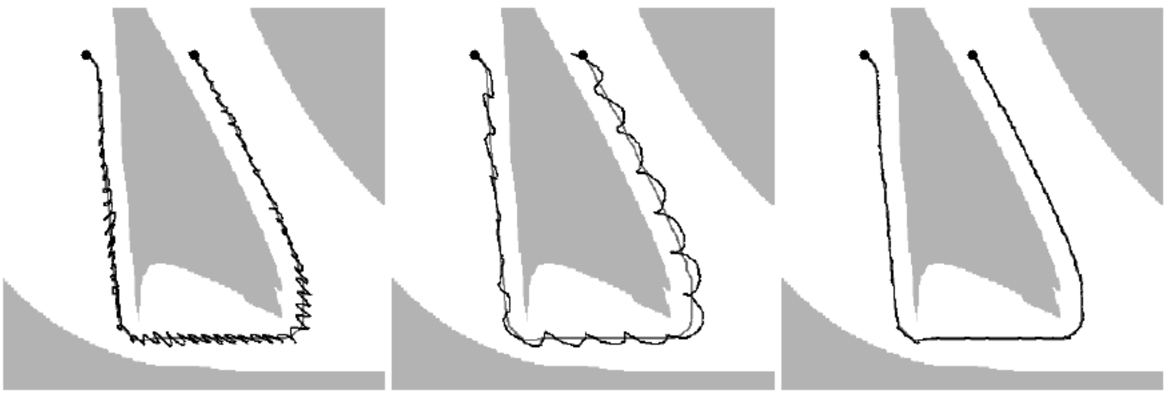
- decreases overshoot
- decreases settling time

PID controller:

$$a_t = K_P(y(t) - x_t) + K_D \frac{\partial(y(t) - x_t)}{\partial t} + K_I \int (y(t) - x_t)\, dt$$

- eliminates steady-state error

Closed-loop control



P control $K_P = 1.0$

P control $K_P = 0.1$

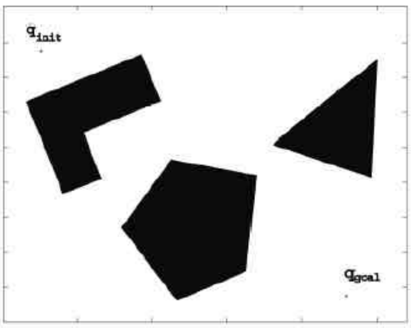
PD control $K_P = 0.3,\ K_D = 0.3$

Potential Field Control

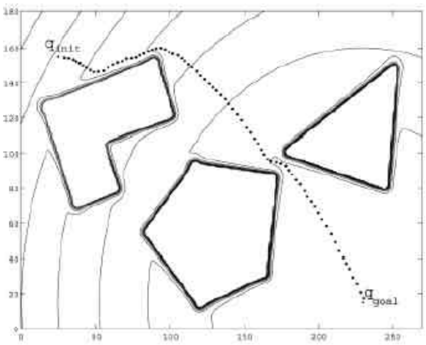
- Motion planning: pot. field as additional cost function
- Generating robot motion directly: superpose potential field of obstacles with field generated by an attractive force, follow the gradient
- Efficient calculation possible, especially compared to path planning
- Problem: local minima, only kinematic method

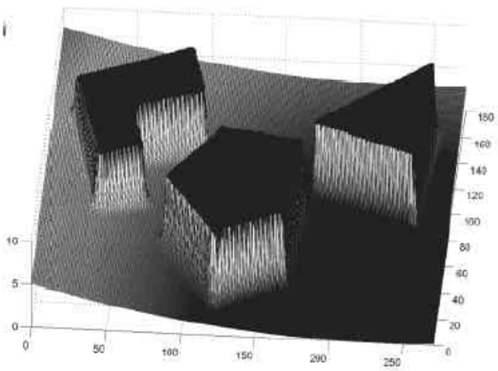
Video

Potential Field Control



- Robot is treated as a *point under the influence* of an artificial potential field.
- Generated robot movement is similar to a ball rolling down the hill
  - Goal generates attractive force
  - Obstacle are repulsive forces





Siegwart, I. Nourbakhsh

Reactive Control

Before: model of environment required (reference path, potential field)

Problems with model based approaches:

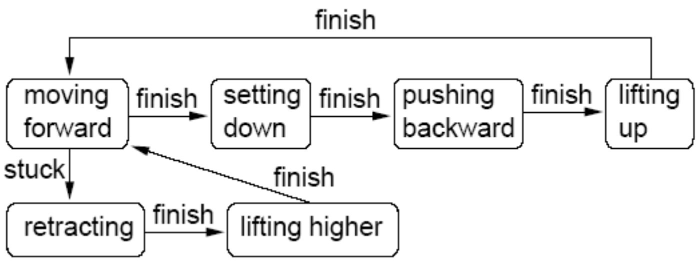
- accurate models are difficult to obtain, especially in complex or remote environments
- computational difficulties, localization errors

→ reactive control

Event driven systems (↔transformational systems) →finite state machines

Environment feedback plays crucial role in the generated behavior

Reactive Control



Emergent behavior: behavior that emerges through interplay of (simple) controller and (complex) environment

Characteristic of great number of biological organisms

Video

Mariusz Wzorek

Artificial Intelligence & Integrated Computer Systems Division
Department of Computer and Information Science
Linköping University, Sweden