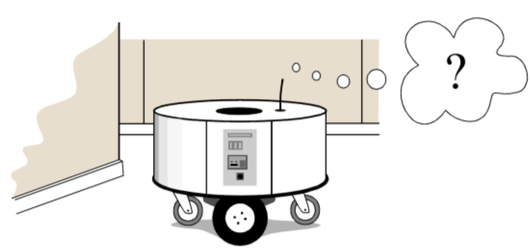
Sistemi Intelligenti Avanzati
Corso di Laurea in Informatica, A.A. 2024-2025
Università degli Studi di Milano

Introduction to Autonomous Mobile Robotics

Michele Antonazzi
Dipartimento di Informatica
michele.antonazzi@unimi.it

Lessons

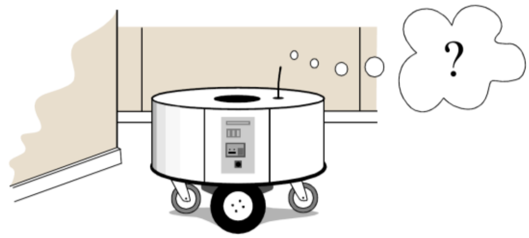
- 09/12/2024 (Antonazzi): Introduction to Autonomous Mobile Robotics
- 12/12/2024 (Antonazzi): Programmazione ROS (in AISLab)
- 16/12/2024 (Antonazzi): Robotic Vision
- 19/12/2024 (Brambilla & Ligabue): Affective Computing
- 09/01/2025 (Luperto): Ricerca su Albero



From [Siegwart, Introduction to Autonomous Mobile Robots]

Outline

From [Siegwart, Introduction to Autonomous Mobile Robots]



- **Introduction**
- Robot Motion
- Perception
- Localization and Mapping
- Navigation

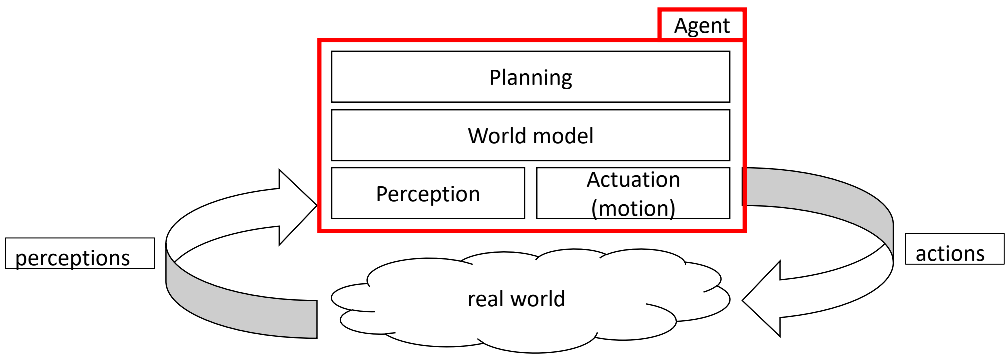
Assumption: let's talk about the simplest type of mobile robots, wheeled ground vehicles

Autonomous Mobile Robots

An agent that autonomously moves inside a given environment, to perform a given task

Autonomous Mobile Robots

"[…] a computer system that is situated in some ***environment,*** *and that is capable of* ***autonomous action*** *in this environment in order to meet its delegated objectives." [Wooldrige, 2009]*



Examples: Manipulators

- Manipulators perform repetitive simple tasks into controlled environments
- Despite their high costs, manipulators are widely used in manufacturing for performing repetitive tasks

Examples: Domestic Robots

Domestic robots are slowly becoming a common item in our homes, but even in this case they have limited abilities and they can perform only simple tasks (vacuum cleaners, lawnmowers, …)

Examples: Collaborative Robots

Collaborative robots are domestic robots that help humans in performing more complex tasks:

- Patrolling
- Objects finding and Graspring
- Healthcare

Examples: Autonomous Driving Cars

Autonomous driving cars are “almost” here, however:

- Driving in roads is a problem that is “easy” to be modeled
- How to do the last mile towards *really* having autonomous road vehicles is still unknown

Limitations of Autonomous Robots




Broadly speaking: if we simplify the environment enough, and we simplify the robot’s tasks enough, we can *have* autonomous robots…

…but there are still major limitations that prevents the widespread adoption of such machines.

Limitations of Autonomous Robots

An agent that autonomously moves inside a given environment, to perform a given task

The major limitations regard the fact than robots need to make *decisions* to adapt their behaviour to the *environment* towards reaching their *tasks:*

- *Embodiment* = is it related to limitation in the robot HW?
- *Cognition* = is it related to limitation in the robot reasoning / SW?

Limitations of Autonomous Robots

Limitations of Autonomous Robots

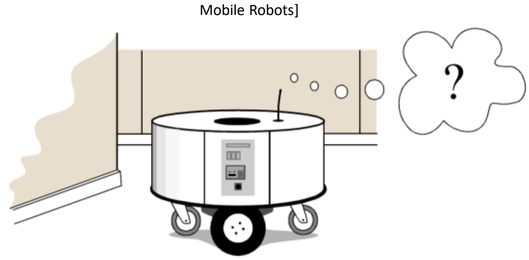
An agent that autonomously moves inside a given environment, to perform a given task

- It seems that, while we still have limitations in terms of robots' actuation, and computational power, the main limitation is still related to their cognition level, i.e., how to make decisions.
- The main one is ***perception***, as it involves the *interpretation* of sensed data in a meaningful way.

Outline

- Introduction
- **Robot Motion**
- Perception
- Localization and Mapping
- Navigation

From [Siegwart, Introduction to Autonomous Mobile Robots]



Assumption: let's talk about the simplest type of mobile robots, wheeled ground vehicles

Robot Wheels

1

2

3

4

Four main types of wheels:

1. **Standard wheel** – 2 DOF – rotation around the wheel axle
2. **Castor wheel** – 2 DOF – rotation around the steering joint
3. **Swedish wheel** – 3 DOF – rotation around wheel axle, rollers, contact point, 45° or 90°
4. **Ball or Spherical Wheel**

Swedish wheels = omnidirectional

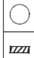
Wheels Configuration

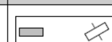
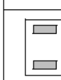
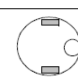
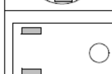
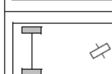
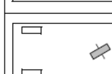
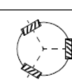
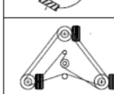
The fundamental characteristics of a robot's locomotion system are:

- **Stability**: stability requires at least two wheels while three wheels ensures static (or passive) stability
- **Maneuverability**: the range of directions that a robot can follow
- **Controllability:** the difficulty in controlling the movements

Usually, maneuverability and controllability are inversely correlated

Wheels Configuration

Icons for the each wheel type are as follows:	
	unpowered omnidirectional wheel (spherical, castor, Swedish);
	motorized Swedish wheel (Stanford wheel);
	unpowered standard wheel;
	motorized standard wheel;
	motorized and steered castor wheel;
	steered standard wheel;
	connected wheels.

# of wheels	Arrangement	Description	Typical examples
2		One steering wheel in the front, one traction wheel in the rear	Bicycle, motorcycle
		Two-wheel differential drive with the center of mass (COM) below the axle	Cye personal robot
3		Two-wheel centered differential drive with a third point of contact	Nomad Scout, smartRob EPFL
		Two independently driven wheels in the rear/front, 1 unpowered omnidirectional wheel in the front/rear	Many indoor robots, including the EPFL robots Pygmalion and Alice
		Two connected traction wheels (differential) in rear, 1 steered free wheel in front	Piaggio minitrucks
		Two free wheels in rear, 1 steered traction wheel in front	Neptune (Carnegie Mellon University), Hero-1
		Three motorized Swedish or spherical wheels arranged in a triangle; omnidirectional movement is possible	Stanford wheel Tribolo EPFL, Palm Pilot Robot Kit (CMU)
		Three synchronously motorized and steered wheels; the orientation is not controllable	"Synchro drive" Denning MRV-2, Georgia Institute of Technology, I-Robot B24, Nomad 200

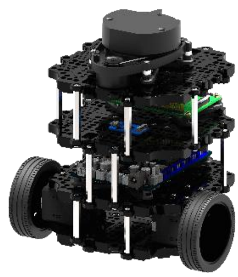
From [Siegwart, Introduction to Autonomous Mobile Robots]

Wheels Configuration

Three wheels robots:

1. Popular configuration
2. Simple
3. High maneuverability, controllability

	Icons for the each wheel type are as follows:
	unpowered omnidirectional wheel (spherical, castor, Swedish);
	motorized Swedish wheel (Stanford wheel);
	unpowered standard wheel;
	motorized standard wheel;
	motorized and steered castor wheel;
	steered standard wheel;
	connected wheels.



# of wheels	Arrangement	Description	Typical examples
2		One steering wheel in the front, one traction wheel in the rear	Bicycle, motorcycle
		Two-wheel differential drive with the center of mass (COM) below the axle	Cye personal robot
3		Two-wheel centered differential drive with a third point of contact	Nomad Scout, smartRob EPFL
		Two independently driven wheels in the rear/front, 1 unpowered omnidirectional wheel in the front/rear	Many indoor robots, including the EPFL robots Pygmalion and Alice
		Two connected traction wheels (differential) in rear, 1 steered free wheel in front	Piaggio minitrucks
		Two free wheels in rear, 1 steered traction wheel in front	Neptune (Carnegie Mellon University), Hero-1
		Three motorized Swedish or spherical wheels arranged in a triangle; omnidirectional movement is possible	Stanford wheel Tribolo EPFL, Palm Pilot Robot Kit (CMU)
		Three synchronously motorized and steered wheels; the orientation is not controllable	"Synchro drive" Denning MRV-2, Georgia Institute of Technology, I-Robot B24, Nomad 200

From [Siegwart, Introduction to Autonomous Mobile Robots]

Wheels Configuration

Three Swedish wheels:

1. Three motors
2. Simple architecture

	Icons for the each wheel type are as follows:
	unpowered omnidirectional wheel (spherical, castor, Swedish);
	motorized Swedish wheel (Stanford wheel);
	unpowered standard wheel;
	motorized standard wheel;
	motorized and steered castor wheel;
	steered standard wheel;
	connected wheels.

# of wheels	Arrangement	Description	Typical examples
2		One steering wheel in the front, one traction wheel in the rear	Bicycle, motorcycle
		Two-wheel differential drive with the center of mass (COM) below the axle	Cye personal robot
3		Two-wheel centered differential drive with a third point of contact	Nomad Scout, smartRob EPFL
		Two independently driven wheels in the rear/front, 1 unpowered omnidirectional wheel in the front/rear	Many indoor robots, including the EPFL robots Pygmalion and Alice
		Two connected traction wheels (differential) in rear, 1 steered free wheel in front	Piaggio minitrucks
		Two free wheels in rear, 1 steered traction wheel in front	Neptune (Carnegie Mellon University), Hero-1
		Three motorized Swedish or spherical wheels arranged in a triangle; omnidirectional movement is possible	Stanford wheel Tribolo EPFL, Palm Pilot Robot Kit (CMU)
		Three synchronously motorized and steered wheels; the orientation is not controllable	"Synchro drive" Denning MRV-2, Georgia Institute of Technology, I-Robot B24, Nomad 200

From [Siegwart, Introduction to Autonomous Mobile Robots]

Wheels Configuration

Car configuration:

1. High controllability
2. Low maneuverability
3. High stability at high velocity

# of wheels	Arrangement	Description	Typical examples
4		Two motorized wheels in the rear, 2 steered wheels in the front; steering has to be different for the 2 wheels to avoid slipping/skidding.	Car with rear-wheel drive
		Two motorized and steered wheels in the front, 2 free wheels in the rear; steering has to be different for the 2 wheels to avoid slipping/skidding.	Car with front-wheel drive
		Four steered and motorized wheels	Four-wheel drive, four-wheel steering Hyperion (CMU)
		Two traction wheels (differential) in rear/front, 2 omnidirectional wheels in the front/rear	Charlie (DMT-EPFL)
		Four omnidirectional wheels	Carnegie Mellon Uranus
		Two-wheel differential drive with 2 additional points of contact	EPFL Khepera, Hyperbot Chip
		Four motorized and steered castor wheels	Nomad XR4000

Icons for the each wheel type are as follows:	
	unpowered omnidirectional wheel (spherical, castor, Swedish);
	motorized Swedish wheel (Stanford wheel);
	unpowered standard wheel;
	motorized standard wheel;
	motorized and steered castor wheel;
	steered standard wheel;
	connected wheels.

From [Siegwart, Introduction to Autonomous Mobile Robots]

Wheels Configuration

Four Swedish wheels

1. High maneuverability
2. Low controllability
3. Omnidirectional

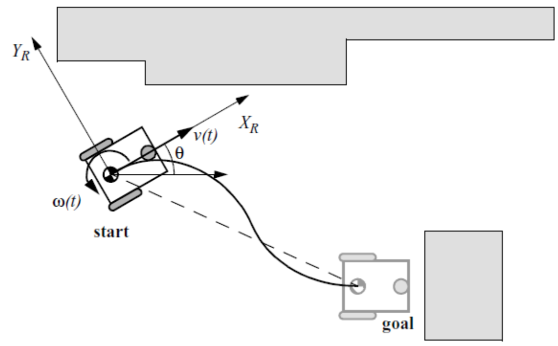
# of wheels	Arrangement	Description	Typical examples
4		Two motorized wheels in the rear, 2 steered wheels in the front; steering has to be different for the 2 wheels to avoid slipping/skidding.	Car with rear-wheel drive
		Two motorized and steered wheels in the front, 2 free wheels in the rear; steering has to be different for the 2 wheels to avoid slipping/skidding.	Car with front-wheel drive
		Four steered and motorized wheels	Four-wheel drive, four-wheel steering Hyperion (CMU)
		Two traction wheels (differential) in rear/front, 2 omnidirectional wheels in the front/rear	Charlie (DMT-EPFL)
		Four omnidirectional wheels	Carnegie Mellon Uranus
		Two-wheel differential drive with 2 additional points of contact	EPFL Khepera, Hyperbot Chip
		Four motorized and steered castor wheels	Nomad XR4000

Icons for the each wheel type are as follows:	
	unpowered omnidirectional wheel (spherical, castor, Swedish);
	motorized Swedish wheel (Stanford wheel);
	unpowered standard wheel;
	motorized standard wheel;
	motorized and steered castor wheel;
	steered standard wheel;
	connected wheels.

From [Siegwart, Introduction to Autonomous Mobile Robots]

Kinematics

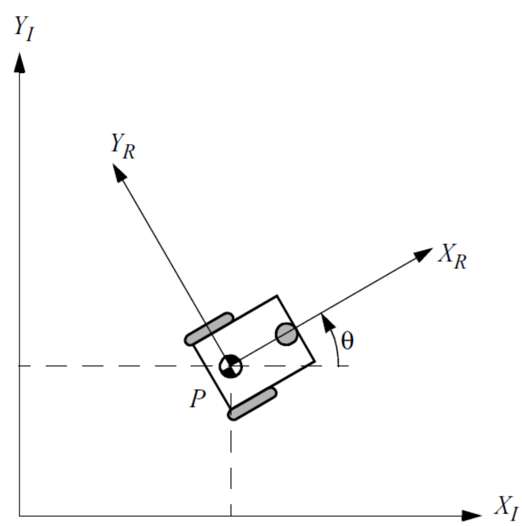
- Describes how a mechanical system behaves
- **Forward Kinematics** computes the robot trajectory in the globa reference frame given the spinning speed of each wheel (localization)
- **Inverse Kinematics** compute the robot actuators parameters to reach a given configuration (control software)



From [Siegwart, Introduction to Autonomous Mobile Robots]

Kinematics

- A crucial challenge in mobile robotics in the **position estimation**
- Kinematic Model of the robot represent the robot position and the robot movement in a global and local reference frame

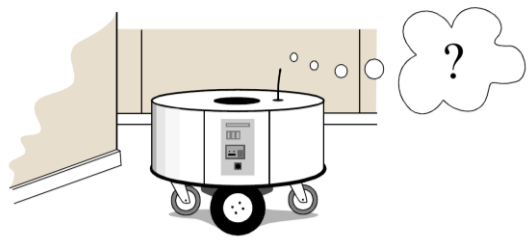


From [Siegwart, Introduction to Autonomous Mobile Robots]

Outline

- Introduction
- Robot Motion
- **Perception**
- Localization and Mapping
- Navigation

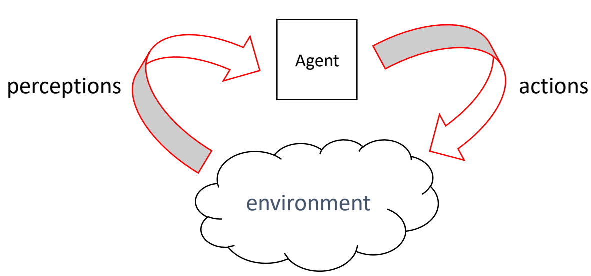
From [Siegwart, Introduction to Autonomous Mobile Robots]



Assumption: let's talk about the simplest type of mobile robots, wheeled ground vehicles

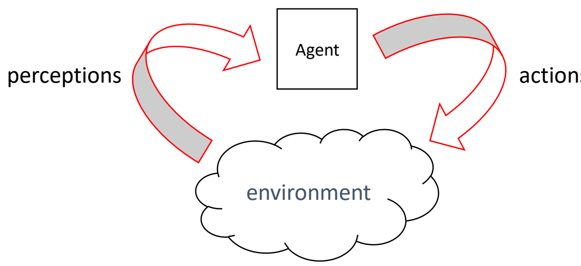
Sensors types

- **Proprioceptive sensors:** measure values internal to the system as motor speed, wheel load, robot arm joint angles, battery voltage
- **Exteroceptive sensors:** acquire information from the robot's environment as distance measurement, light intensity, sound amplitude = meaningful environmental features



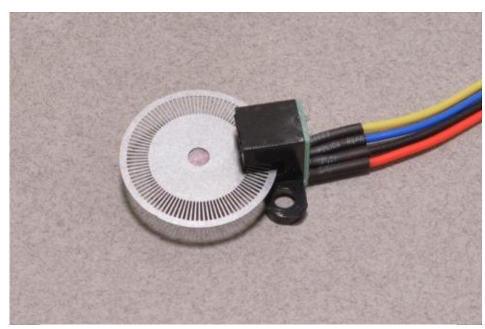
Sensors types

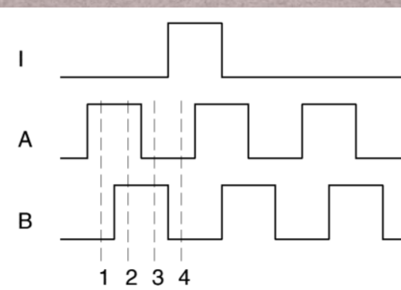
- **Passive sensors:** measure ambient environmental energy entering the sensors, as microphones, temperature probes, cameras
- **Active sensors:** emit energy into the environment, then measure the environmental reaction. More controllable, more accurate, but interference issues (and sometimes power)



Wheel/Motor Sensors

- Proprioceptive sensors
- **Optical encoders**: measure the angular speed and position in a motor drive or steering mechanism
- Used to estimate the robot movements (localization)
- **Odometry** is the use of data from motion sensors to estimate change in position over time





Heading Sensors

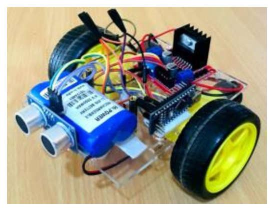
- They describe the robot's orientation and inclination
- Compasses: outdoor
- Inertial Measurements Unit (IMU):
  - Accelerometers + gyroscopes
  - Measures the relative position, acceleration, and position of a moving device
  - Subject to drift
- Beacons:
  - Active or passive: RFID, NFC, Bluetooth, markers, etc
  - GPS: performs poorly in indoor applications

Active Ranging Sensors

- Most popular sensors in mobile robotics
- Provide direct measurements of distance from the robot to objects in its vicinity
- Among them, *time-of-flight* sensors are those commonly used
  - $d = c \cdot t$
  - d: distance travelled
  - c: speed of wave propagation
  - t: time of flight

Ultrasonic Sensors - Sonars

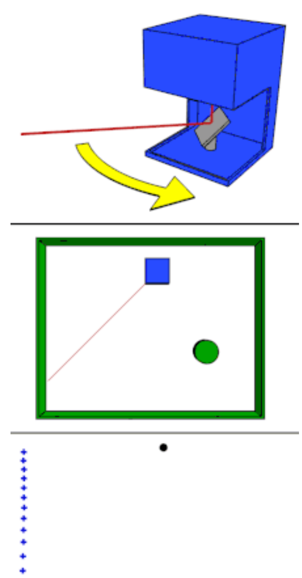
- They emit ultrasonic waves
- Advantages:
  - Cheap
  - Good for obstacle avoidance
  - Simple and interpretable measurements
- Disadvantages:
  - Not particularly accurate
  - Narrow measurement area
  - Low range



Laser Range Finders - Lidars

Widely used in most indoor and outdoor robot applications as they:

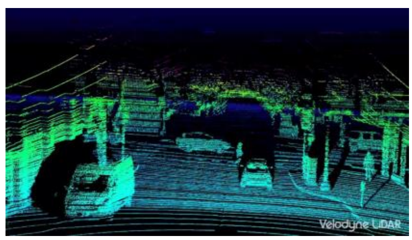
- Are relatively cheap
- Easy to use and provide interpretable measures
- Robust wrt environmental changes (e.g., day, night, different seasons)



Laser Range Finders - Lidars

- Range = max perceivable distance (1-100m)
- Field of View (FOV) = degrees of a scan, from 180° to 360°
- Angular resolution = how many points for each degree in a scan
- Frequency = how many scans per second (1 Hz – 50 Hz)

Indoor 2D lidar



Outdoor 3D lidar

Sensors for Vision

Cameras, by acquiring visual data, enable the robot to solve high-level tasks (also thanks to Deep Learning).
Drawbacks:

- Images are difficult to interpret
- Limited range
- *Reliability* (day-night or light changes)

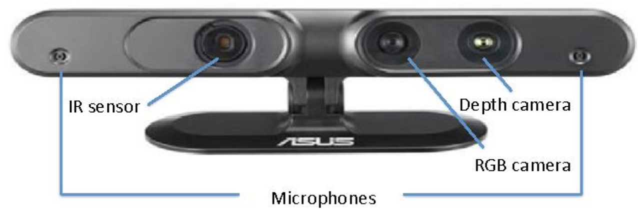
RGBD Cameras

They acquire RGB + depth information

- Allow to easily reconstruct 3D image of the environment
- Widely used and useful, especially indoor (obstacles avoidance, human detection, etc…)
- Cheap (100€→1000€)

Drawbacks:

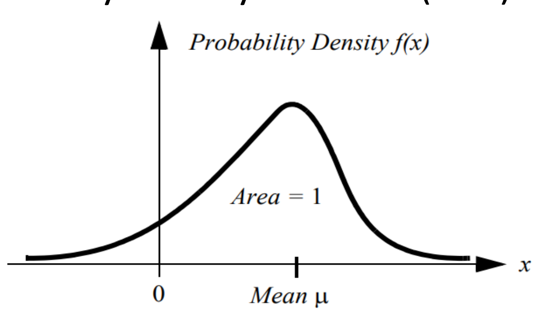
- Limited range - depth (range < 3/5m)
- Do not replace vision



(b) Asus Xtion Sensor

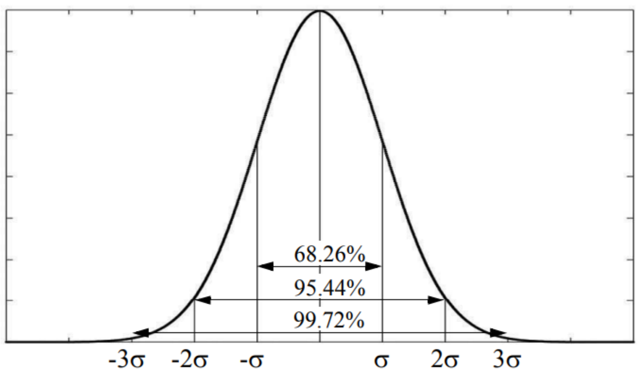
Representing Uncertainty

- Sensors are imperfect devices with systematic and random error
- We need a tool for modelling and treat the sensors' uncertainty
- Considering a set of measurements *n* which values p_i, our goal is to estimates $E[X] = g(p_1, p_2, p_3, ..., p_n)$
- We can use a probability density function (PDF) to characterize *X*



Uncertainty as Guassian Distribution

- The Gaussian's PDF depends only on μ and σ
- It is symmetric around μ
- It has tails that approach to zero

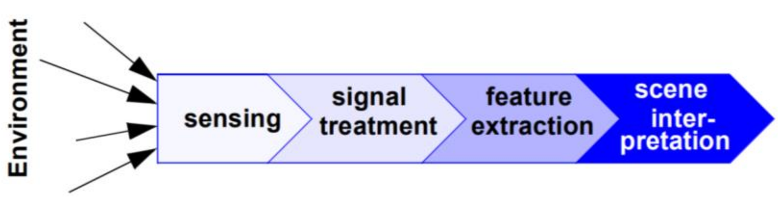


$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Feature Extraction

How can the robot use input sensor values?

- Consider each raw sensor measurements as an individual value
- Build and update an *high-level* model from values of one or more sensors **(feature extraction).** Features are abstraction of raw data and can be:
  - **Low level features:** corners, edges, lines, …
  - **High level features:** objects, semantic labels, scene understanding, …



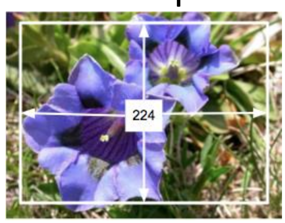
From [Siegwart, Introduction to Autonomous Mobile Robots]

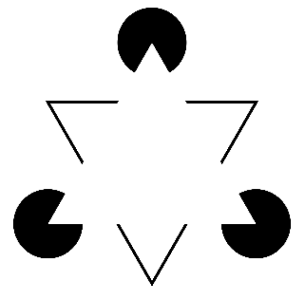
Feature Extraction and Computation

- **Features extraction:** a way to extract *a few meaningful information* from a *dense and complex* input
- We are able to identify features/pattern easily, robots don't

E.g. an 8K video streams has a bandwidth of 300Mb/s

- How much of that info is needed by the robot?
- How much of that info can be processed in real time?

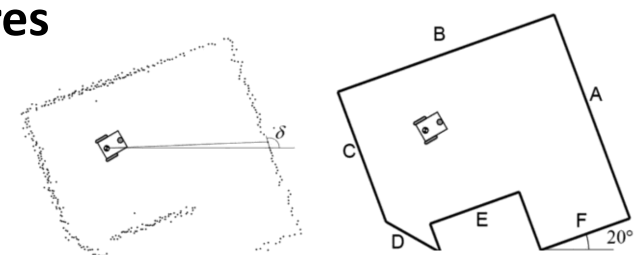




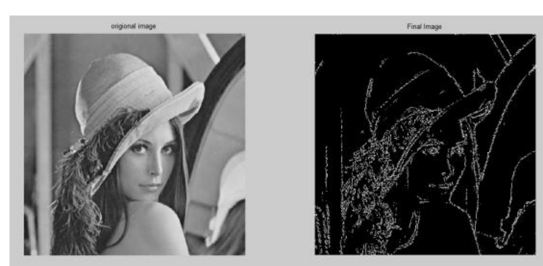
Desired Property of Features

Features should be:

- Mathematically described
- Always perceivable and easily detectable (by humans)
- Localized in the environment model
- Invariant with respect to viewpoint, illumination, scale
- Computationally efficient and robust (artifacts, noise, or distortions should not affect the feature detection)



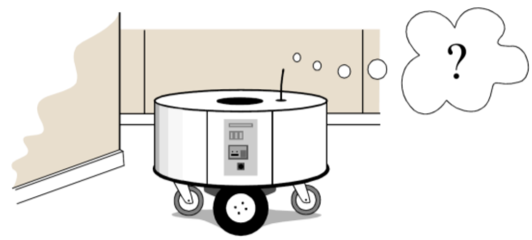
Line extraction in range data



Edge detection in visual data

Outline

From [Siegwart, Introduction to Autonomous Mobile Robots]



- Introduction
- Robot Motion
- Perception
- **Localization and Mapping**
- Navigation

Assumption: let's talk about the simplest type of mobile robots, wheeled ground vehicles

Challenges of Robot Mobility

Robot mobility requires addressing a key property: ***uncertainty***

- **Real world environments** are unpredictable
- **Sensors** are subjects to noise and errors
- **Robots actuation** is unpredictable, an action can not have the desired effect
- **Environment models** are inherently inaccurate (they are abstraction)
- **Real-time computation** is often approximated

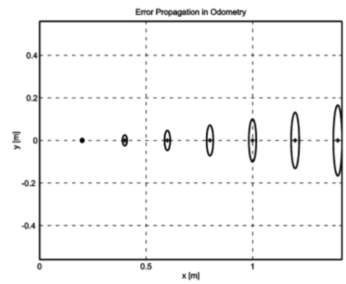
Pose Estimation with Odometry

Odometry is the use of data from motor sensors to estimate change in position by integrating the movements over time

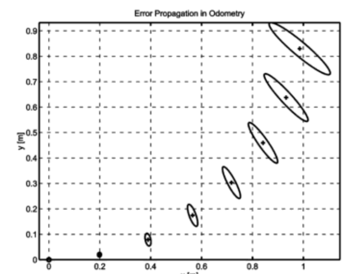
Motion model: $p\left(x_{r,t} \middle| x_{r,t-1}, u_t\right)$

Odometric position updates can give only a very rough estimate of the actual robot's:

- Integration errors
- Motion errors:
  - Misalignment of wheels
  - Unequal wheel diameter
  - Variation in the contact point of the wheels
  - Irregular surfaces



Growth of the pose uncertainty in straight-line and circular movements



From [Siegwart, Introduction to Autonomous Mobile Robots]

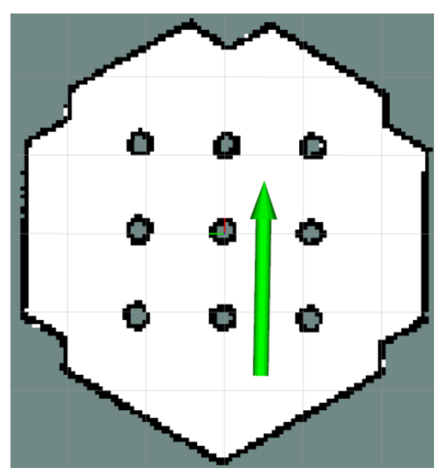
Pose Estimation using Map

Sensor aliasing = nonuniqueness of sensor readings

- In Robotics, there is a many-to-one mapping from environmental states to the robot's perceptual inputs. The robot cannot distinguish different states.

The human sensory system, particularly the visual system, tends to receive unique inputs in each unique local state

- We experience aliasing in unfamiliar context: total dark, mazes, environments without landmarks.



Combining Odometry and Map for Localization

The robot performs localization by collecting sensor data and updating some **belief** about its position with respect to the environment map

Advantages of this approach:

- Allows to use exteroceptive sensors
- Makes the belief transparent to humans
- The map (built by the robot) can be used also by humans
- The robot can localize itself in a new environment with a new map

From [Siegwart, Introduction to Autonomous Mobile Robots]

Belief for Localization

The robot **belief** is a probability distribution over the space of all possible locations of the current robot pose

$$bel_t(x_r) = p(\, x_{r,t} | z_{1:t}, u_{0:t})$$

Where $z_{1:t}$ are exteroceptive sensor readings and $u_{0:t}$ sequence of proprioceptive data from motor sensors

- **Motion model:**

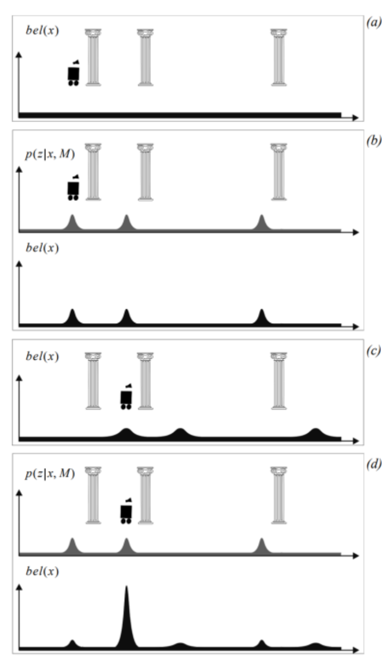
$$p(x_t | x_{r,t-1}\, , u_{r,t}\,)$$

describes the probability that the robot position is x_t given it previous state (x_{t-1}) and control command $u_{r,t}$,

- **Measurements model:**

$$p(z_t | x_{r,t}, M)$$

describes the probability of a robot measurement z_t given a robot pose $x_{r,t}$ given a map M



From [Siegwart, Introduction to Autonomous Mobile Robots]

Map Representations

Property of the map:

1. The map's precision reflects the localization granularity
2. The map's representation matches the data types returned by sensors
3. The complexity of the map representation has direct impact on the computational complexity of reasoning about mapping, localization, and navigation

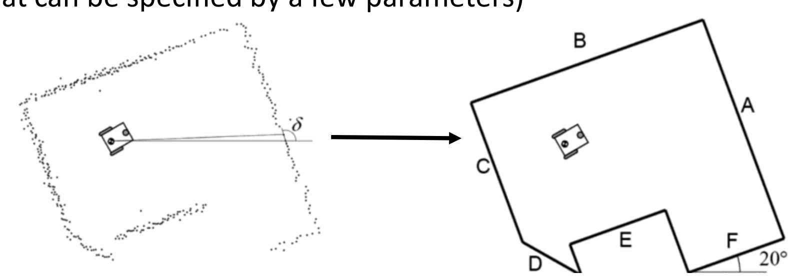
Continuous Representation

Method for *exact* decomposition of the environment

Pros: high accuracy with respect to the robot position.

Cons: too costly (memory and computational time)

1. **The closed-world assumption:** we store in the map only the obstacles
2. **Feature extraction:** the robot extracts best-fit lines from the thousands of points of lidar (that can be specified by a few parameters)



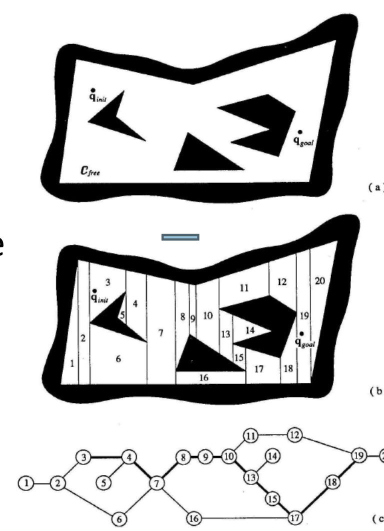
From [Siegwart, Introduction to Autonomous Mobile Robots]

Environment Decomposition

- This technique decomposes the environment in sub-regions, producing an abstraction of the real-world.
- **Disadvantage:** loss of fidelity between the map and the real world (both qualitatively and quantitatively)
- **Advantages:**
  - Allows to capture the *useful* features of the world
  - Decomposition can be *hierarchical* according to the desired task
  - The reasoning and planning on a simplified map is computationally efficient

Exact Cell Decomposition

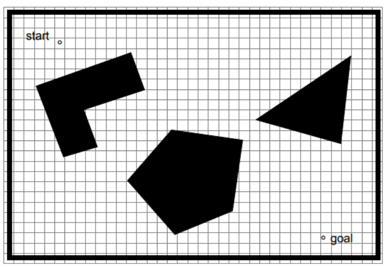
- This method use critical points to tesselate environment, obtaining a discrete topological map from a continuous one
- The representation can be extremely compact because each such area is stored as a single node
- **Assumption:** the precise position of the robot within each sub-portion does not matter, what matters is the ability of the robot to move from area to area.

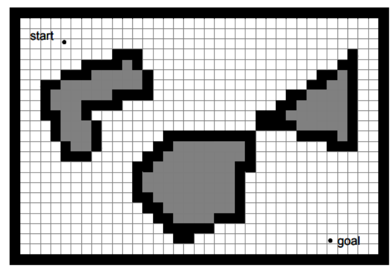


From [Siegwart, Introduction to Autonomous Mobile Robots]

Fixed Decomposition

- This method (extremely popular) discretizes the environment in a map divided into equal cells
- **Compact representation:** the map can be represented as a matrix, in which each cell can be free or occupied {0, 1} (**grid map**)
- By assigning different values, e.g. in [0, 1], we can define the occupancy probability of each cell (**occupancy grid map**)



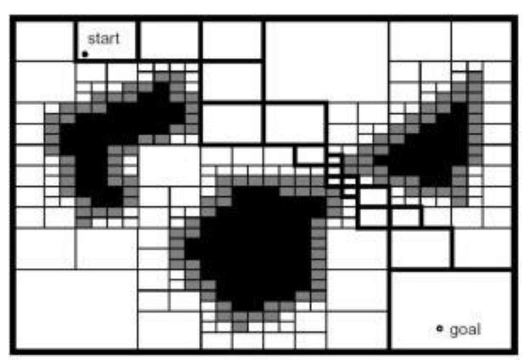


From [Siegwart, Introduction to Autonomous Mobile Robots]

Approximate Decomposition

Cells can have different sizes

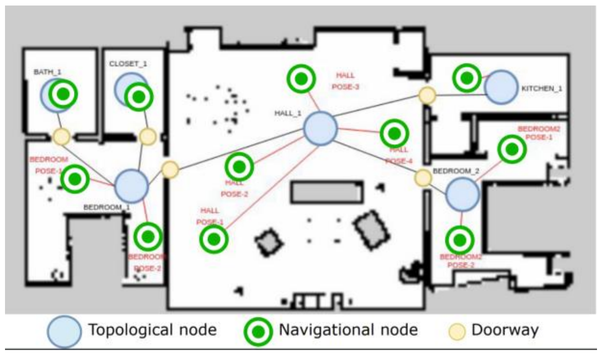
- Particularly suited for sparse environments
- Heps to reduce the complexity and the memory usage (especially in 3D grid maps)



From [Siegwart, Introduction to Autonomous Mobile Robots]

Topological Maps

- A topological map is a graph composed by nodes (different locations) and arcs (direct connectivity between two locations)
- Generally, topological maps are combined with grid maps for solving different task



The Map-based Localization Problem

The robot map-based localization is divided into two steps:

- **Prediction (or action) update:** the robot uses its proprioceptive sensors to estimate its position. In this phase the *uncertainty increases* during time
- **Perception (or correction) update:** the robot uses its exteroceptive sensors to localize itself with respect to a map, this adjusts the position estimated in the previous step. In this phase, the uncertainty shrinks

Map-based Localization Methods

- **Markov localization:**
  - The belief is represented using any arbitary PDF, considering every possible position
  - It requires a *discrete* representation of the space (e.g., grid maps)
- **Kalman filter localization:**
  - The belief is represented as simple Gaussian PDF (only mean and covariance are updated during localization)
  - Suitable for *continous* world representations

Markov Localization

- **Markov assumption:** the belief x_t depends only on its previous state $x_{t-1}$the most recent odometry u_t and perception z_t values
- The robot position space is discretized in a finite number of poses (x, y, θ)

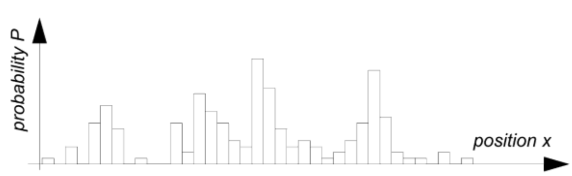
Two phases:

1. **Prediction (action) update:**

$$\boldsymbol{bel(x_t) = \sum_{x_{t-1}} p(x_t|u_t, x_{t-1})bel(x_{t-1})}$$

2. **Perception (measurement) update:**

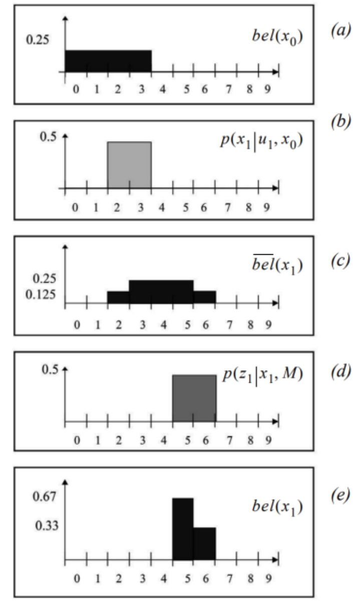
$$\boldsymbol{bel(x_t) = np(z_t|x_t, M)bel(x_t)}$$



From [Siegwart, Introduction to Autonomous Mobile Robots]

Markov Localization Example

a) The robot belief is uniformelly distributed
b) Between t = 0 and t = 1, the robot may have moved either two or three cells
c) The new belief at t = 1, calculated using the **motion model**, is given by the sum of:
   - $p(x_1 = 2) = p(x_0 = 0)p(u_1 = 2) = 0.125$
   - $p(x_1 = 3) = p(x_0 = 0)p(u_1 = 3) + p(x_0 = 1)p(u_1 = 2) = 0.25$
   - $p(x_1 = 4) = p(x_0 = 1)p(u_1 = 3) + p(x_0 = 2)p(u_1 = 2) = 0.25$
   - $p(x_1 = 5) = p(x_0 = 2)p(u_1 = 3) + p(x_0 = 3)p(u_1 = 2) = 0.25$
   - $p(x_1 = 6) = p(x_0 = 3)p(u_1 = 3) = 0.125$
d) The robot, using sensors, measures that distance from the origin can be equally 5 or 6 cells
e) The belief is updated (and fixed) using the **perception model:**
   - $p(x_1 = 5) = p(x_5 = 0.25)p(z_1 = 5) = 0.125$
   - $p(x_1 = 6) = p(x_6 = 0.125)p(z_1 = 6) = 0.0625$
   - The normalization constant $\mathrm{n} = \frac{1}{0.125+0.0625} \cong 5.33$
   - $\mathrm{n}p(x_1 = 5) = 5.33 \ * 0.125 \cong 0.67$
   - $\mathrm{n}p(x_1 = 6) = 5.33 \ * 0.125 \cong 0.33$



From [Siegwart, Introduction to Autonomous Mobile Robots]

Markov Localization Considerations

Benefits:
- Localization is possible from every unknown starting position
- Ambiguous situations can be recovered

Limitations:
- Treating a complete belief state in Markov Localization is computationally too hard

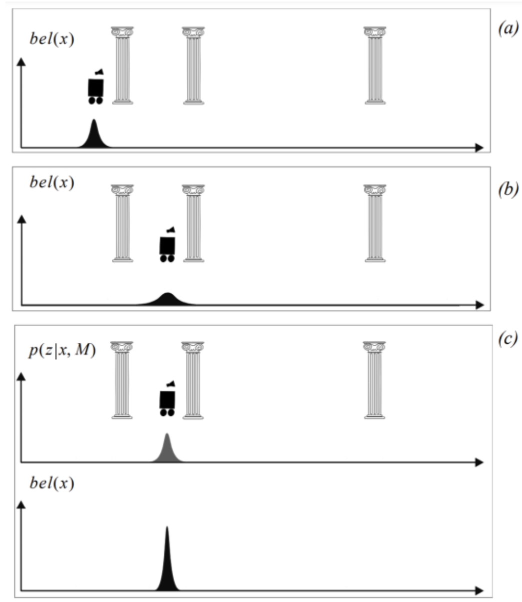
Solution:
- The belief is approximated considering only a subset of possible locations
  - The locations with low probability are discarded
  - This can be done using Particle Filter or Monte Carlo algorithms

Kalman Filter Localization

The belief, the motion model, and the perception model are represented as Gaussian PDFs

Limitations:
- The robot initial pose must be known
- If the uncertainty becomes too high, the localization is impossible

Benefits:
- More efficient than Markov Localization



From [Siegwart, Introduction to Autonomous Mobile Robots]

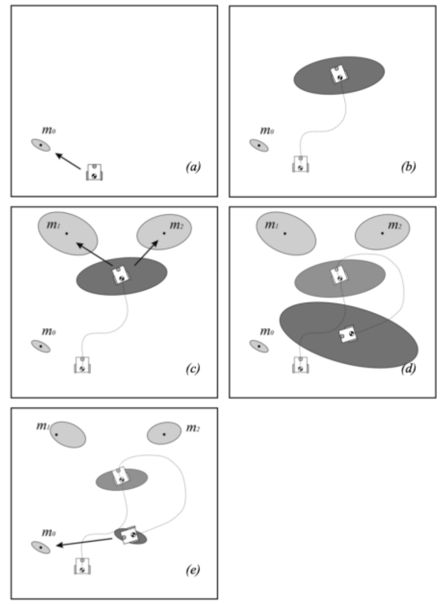
Automatic Map Building

Manually mapping an environment is too difficult and time-consuming:
- The landmarks must be accurately measured
- The look of the map can change according to different perception capabilities
- The environment can dynamically change

The solution is to allow the robot to autonomously build a map of an unknown environment by performing SLAM (Simultaneous Localization and Mapping):
1. Incrementally mapping the environment integrating new observations
2. Localize itself its in the map

The SLAM Problem

- The aim of SLAM is to recover both the robot path and the environment map using proprioceptive and exteroceptive sensor data.
- This is difficult because both the estimated path and the extracted features are corrupted by noise and the uncertainty during mapping incrementally increasing.
- **Loop closure:** the solution is to observe features already observed before (fir which the position is relatively well-known)

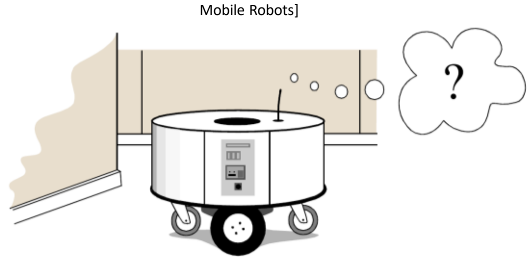


From [Siegwart, Introduction to Autonomous Mobile Robots]

Outline

- Introduction
- Robot Motion
- Perception
- Localization and Mapping
- **Navigation**

From [Siegwart, Introduction to Autonomous Mobile Robots]

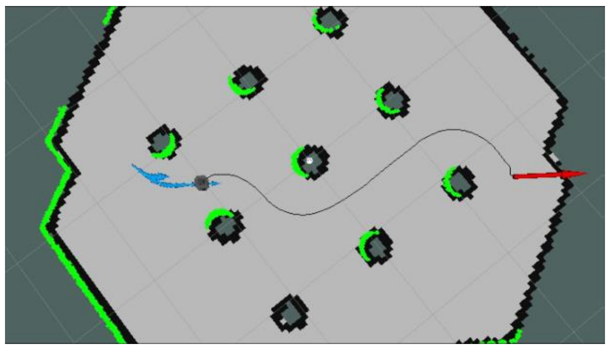


Assumption: let’s talk about the simplest type of mobile robots, wheeled ground vehicles

Navigation

Given a map and a goal position, navigation is the ability of the robot to act based on its knowledge and sensor values to reach such a goal as *efficiently* and as *reliably* as possible.

- **Path Planning:** identify a trajectory to reach the goal
- **Obstacle Avoidance:** modulate the trajectory to avoid collisions



Path Planning Approaches

We have to compute a set of states for finding the path that the robot can execute. Proper formulations for this problem are:

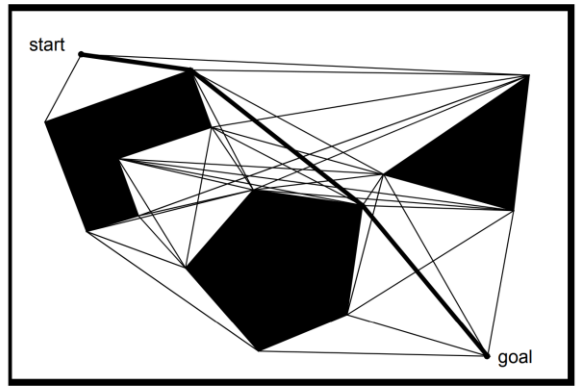
1. **Graph search:** a connectivity graph in free space is first constructed and then searched.
2. **Potential field planning:** a mathematical function is imposed directly on the free space. The gradient of this function can then be followed to the goal.

Graph Construction: Visibility Graph

Graph structure:
- The nodes are poligons' vertices and the start and goal positions
- The edges connects all pair of vertices *that can see each other*

Pros:
- Simple implementation
- Extremely fast and efficient
- Shortest solutions are optimal in terms of path length

Cons: the path is too close to the obstacles



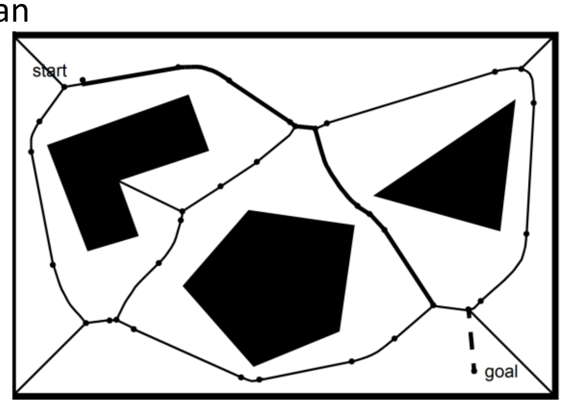
From [Siegwart, Introduction to Autonomous Mobile Robots]

Graph Construction: Voronoi Graph

Graph structure:
- The nodes are points in the free space than maximize the distance to obstacles
- The edges connect these points

Disadvantages:
- The paths are far from optimal solutions
- The localization is in danger if the robot uses shor-range sensors

Advantages:
- **Executability**: The robot can easily follow the Voronoi edges by maximizing the sensors' readings
- **Safety:** the robot is far away from obstacles



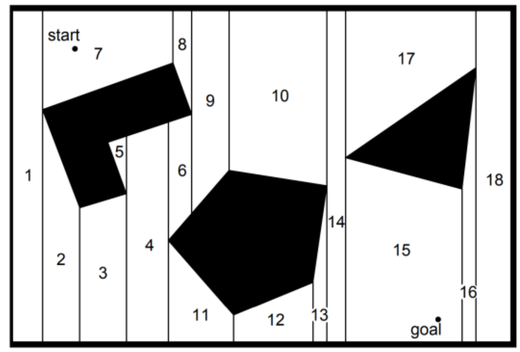
From [Siegwart, Introduction to Autonomous Mobile Robots]

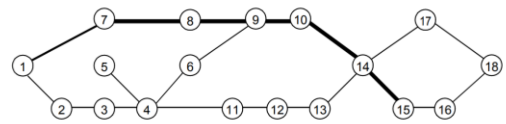
Graph Construction: Exact Cell Decomposition

Graph structure:
- The environment is divided in zones based on geometric criticality
- The nodes represent cells
- The edges connect adjacent cells

Disadvantages:
- Poor localization
- High complexity in cluttered environments

Advantages:
- Low complexity in sparse environments





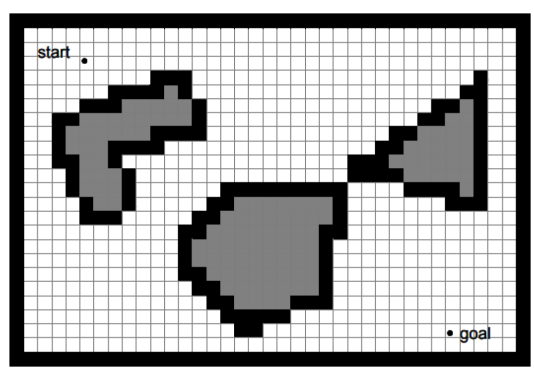
From [Siegwart, Introduction to Autonomous Mobile Robots]

Graph Construction: Approximate Cell Decomposition

The most popular graph construction method in robotics (due to the use of grid maps)

Graph structure:
- The nodes represent each cell
- The edges connect adjacent cells

Advantages:
- Versatility: the cells can have variable sizes
- The great benefit of approximate cell decomposition is the low computational complexity induced to path planning



Potential Field Path Planning

Idea: put an attractive artificial potential field on the goal, a repulsive one on obstacles, let the robot follow these simulated forces