The Navigation Problem

Three main questions
- where am I?
- where am I going?
- how do I get there?

To answer these questions the robot has to
- have a model of the environment
  - given or autonomously built
- perceive and analyze the environment
- find its position within the environment
- plan and execute the environment
**Mapping-Planning-Driving**

- **Mapping**
  - represent the environment where the robot must navigate
    - metric / topological
    - off-line / on-line

- **Planning**
  - consists of building a path that allows to reach the goal

- **Driving**
  - follow the planned path with the right speed, acceleration, and accuracy

These activities may be not sequentially executed
The navigation problem may be faced in
- 2D Cartesian space
- C-space 3D
- C-space 2D
  - for cylindrical and omni-directional robots

To reduce the C-space
- consider the robot as a point
- grow obstacles to avoid collisions
  - the growth depends on the radius of the circle, or on half the diagonal
  - different headings lead to different C-space maps, joined by rotations
- plan the path
The raw sensor data must be processed to extract features. Features are high level information that can be used to model the environment. The features that should be extracted depend on the environment:
- line segments / corners / particular landmarks
The techniques for extracting features depend on the sensor used. Features must be tracked.
Robot Localization

Given the **features** extracted from sensors and the **map** of the environment a robot must identify its own position.

There are several techniques to localize a robot:

- **Incrementally update (dead reckoning)**
  - accumulation of errors
  - not applicable
- **Modifying the environment**
  - requires to structure the environments
  - expensive, inflexible
- **Quantitative metric approach**
  - match features with map elements
  - sensor are noisy
  - probabilistic approaches
  - is the common solution
A Taxonomy of Localization Problems

- Local vs Global Localization
  - position tracking
  - global localization
  - kidnapped robot problem
- Static vs Dynamic Environments
- Passive vs Active
- Single-Robot vs Multi-Robot
Challenges of Localization

- **Sensor noise**
  - due to the environment or the measurement process
  - solution: multiple readings and sensor fusion

- **Sensor aliasing**
  - non-uniqueness of sensor readings
  - in general, robot position may not be identified from a single reading

- **Effector noise**
  - robot movement is perceived through odometric sensors and is integrated
  - easy to do, but errors are integrated $\Rightarrow$ unbound

- **Odometric position estimation**
  - systematic vs non-systematic errors
Error types
- systematic
  - can be eliminated through calibration
- non-systematic
  - require error models, and will always lead to uncertain position estimation

Error sources
- sensor resolution
- misalignment of the wheels
- unequal wheel diameter
- variation in the contact point of the wheel
- unequal floor contact
- ...

Odometry
Odometry: the Differential Drive Robot

\[
p = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \quad p' = p + \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta \theta \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} \Delta s_r + \Delta s_l \\ \Delta s_r + \Delta s_l \\ \Delta s_r - \Delta s_l \end{bmatrix} \begin{bmatrix} \frac{\Delta s_r + \Delta s_l}{2} \cos\left(\theta + \frac{\Delta s_r - \Delta s_l}{2b}\right) \\ \frac{\Delta s_r + \Delta s_l}{2} \sin\left(\theta + \frac{\Delta s_r - \Delta s_l}{2b}\right) \\ \frac{\Delta s_r - \Delta s_l}{b} \end{bmatrix}
\]
Odometry: Growth of Pose Uncertainty
Odometry: Calibration of Errors

a. Preprogrammed square path, 4x4 m.

b. Preprogrammed square path, 4x4 m.

87° turn instead of 90° turn (due to uncertainty about the effective wheelbase).

Curved instead of straight path (due to unequal wheel diameters).
In the example here, this causes a 3° orientation error.

93° turn instead of 90° turn (due to uncertainty about the effective wheelbase).

Preprogrammed square path, 4x4 m.
To Localize or Not?

- Behavior Based vs Model Based Navigation
Belief Representation

- Continuous map with single hypothesis

- Continuous map with multiple hypotheses

- Discretized map with probability distribution

- Discretized topological map with probability distribution
The features extracted are then matched with (or used to update) the environment representation.

The environment may be represented in different ways:

- continuous metric
- discrete metric
- discrete topological
Maps: Some Decompositions

- Topological Decomposition

node
cannectivity
A mobile robot moving in a known environment
Starting from a known location it keeps track of its current location using odometry
- the uncertainty about its location growth
  - update the location using information from the environment
The information extracted from the observation of the environment must be fused with the odometric estimation to achieve the best estimation of the robot's actual position
  - Action update (increase uncertainty)
    \[ s'_t = \text{Act}(o_t, s_{t-1}) \]
  - Perception update (reduce uncertainty)
    \[ s_t = \text{See}(i_t, s'_t) \]
**Probabilistic Map Based Localization: The Process**

**Given**
- the position estimate
- its covariance
- control input
- current observations
- the map

**Compute**
- new (posteriori) position estimate
- its covariance

Diagram:
- Encoder
- Map
- Position and Measurement Prediction
- Estimation (fusion)
- Matching
- Observation with on-board sensors

- matched predictions and observations
- row sensor data or extracted features
Markov vs Kalman Filter Localization

Markov Localization
- localization starting from any unknown position
- recovers from ambiguous situation
- however, to update the probability of all positions within the whole state space at any time requires a discrete representation of the space (grid). The required memory and calculation power can thus become very important if a fine grid is used

Kalman Filter localization
- tracks the robot and is inherently very precise and efficient
- however, if the uncertainty of the robot becomes too large (e.g., collision with an object) the Kalman filter will fail and the position is definitively lost
Markov Localization

- Uses an explicit, discrete representation for the probability of all positions in the space
- The representation is typically a grid or a topological graph with a finite number of possible positions
- During each step all the positions are updated
- SEE: map from a belief state and a sensor input to a refined belief state
  \[
  p(l|i) = \frac{p(i|l)p(l)}{p(i)}
  \]
- ACT: map from a belief state and an action to a new belief state
  \[
  p(l_t|o_t) = \int p(l_t|l'_{t-1}, o_t) p(l'_{t-1}) dl'_{t-1}
  \]
- **Markov assumption:** update depends only on previous state and its most recent actions and perception
Markov Localization: An Example
Kalman Filter Localization

- Is a special case of Markov Localization
- We need a motion and a measurement model
- Let us assume that
  - correspondences are known
  - initial position is known

Control → System → Measuring Devices

System error source → System

System state → Measuring Devices

Measurement error sources → Measuring Devices

Observed measurement → Kalman Filter

Kalman Filter → Optimal estimate of system state
Kalman Filter Localization: An Example
Other (Non Probabilistic) Localization Methods

- Localization based on beacons or landmarks
  - triangulation
- Localization based on active beacons
- Localization based on artificial landmarks
  - bar-code
- Localization based on active landmarks
- Angular histogram
Starting from an arbitrary initial point, a mobile robot should be able to autonomously explore the environment with its on-board sensors, gain knowledge about it, interpret the scene, build an appropriate map and localize itself relative to this map.

SLAM
The Simultaneous Localization And Mapping Problem
How to Establish a Map?

- In some situations we cannot (or do not want to) provide a map “by hand”
  - hard and costly
  - dynamically changing environment
  - different look due to different perception

- Map requirements
  - incorporate new sensations into the existing world model
  - information and procedures for localization
  - information for path-planning and other navigation tasks

- Quality of a map
  - topological correctness
  - metrical correctness
Problems in Map Building

- Map maintaining
  - keeping track of changes in the environment
    - measure of belief of each environment feature

- Representation and reduction of uncertainty
  - probability densities of feature positions
  - additional exploration strategies
$M = \{ \hat{z}_t, \Sigma_t, c_t | (1 \leq t \leq n) \}$

credibility factor $c_t$
Map Representation

- $M$ is a set of $n$ probabilistic feature locations.
- Each feature is represented by the covariance matrix $\Sigma_t$ and an associated credibility factor $c_t$.

$$M = \{\hat{z}_t, \Sigma_t, c_t | 1 \leq t \leq n\}$$

- $c_t$ is between 0 and 1 and quantifies the belief in the existence of the feature in the environment:

$$c_t(k) = 1 - e^{-\left(\frac{n_s - n_u}{a - b}\right)}$$

- $a$ and $b$ define the learning and forgetting rate and $n_s$ and $n_u$ are the number of matched and unobserved predictions up to time $k$, respectively.
Small local errors accumulate to arbitrary large global errors.
This is usually irrelevant for navigation.
However, when closing loops, global error does matter.