

Mobile robot localization

Mauro Dragone
University College Dublin
mauro.dragone@ucd.ie

Objective:

**Determination of the pose (= position + orientation)
of a mobile robot in a known environment in order
to successfully perform a given task**

Summary:

- Robot localization **taxonomy** (4-5)
- Robot localization as an **estimation problem**: the **Bayes filter** (6-17)
- **Observability** issue (18-23)
- Robot description (24)
 - **Motion model**: the differential drive (25-34)
 - **Measurement model**: rangefinder or landmarks (35-39)

- **Robot localization algorithms** (40-42):
 - Gaussian filters: **EKF**, **UKF** (43-60)
 - Non parametric filters: HF, **PF** (61-77)
- Examples (78-104)
- Conclusions (105)
- References (106)

Robot Localization Taxonomy:

- **Position tracking**
- **Global localization**
- **Kidnapped problem**



Increasing degree
of difficulty

Other classifications include:

- type of environment (static or dynamic; completely or partially known)
- active or passive localization
- single or multi robot localization.

Robot Localization Taxonomy:

- Position tracking
- Global localization
- **Kidnapped problem**



Increasing degree
of difficulty

Other classifications include:

- type of environment (static or dynamic; **completely** or partially **known**)
- active or passive localization
- single or multi robot localization.

Localization as an estimation problem

The robot must infer its pose from available data

Data (noisy):

- Motion information:

→ Proprioceptive sensors (e.g. encoders, accelerometers, etc.)

- Environment Measurements

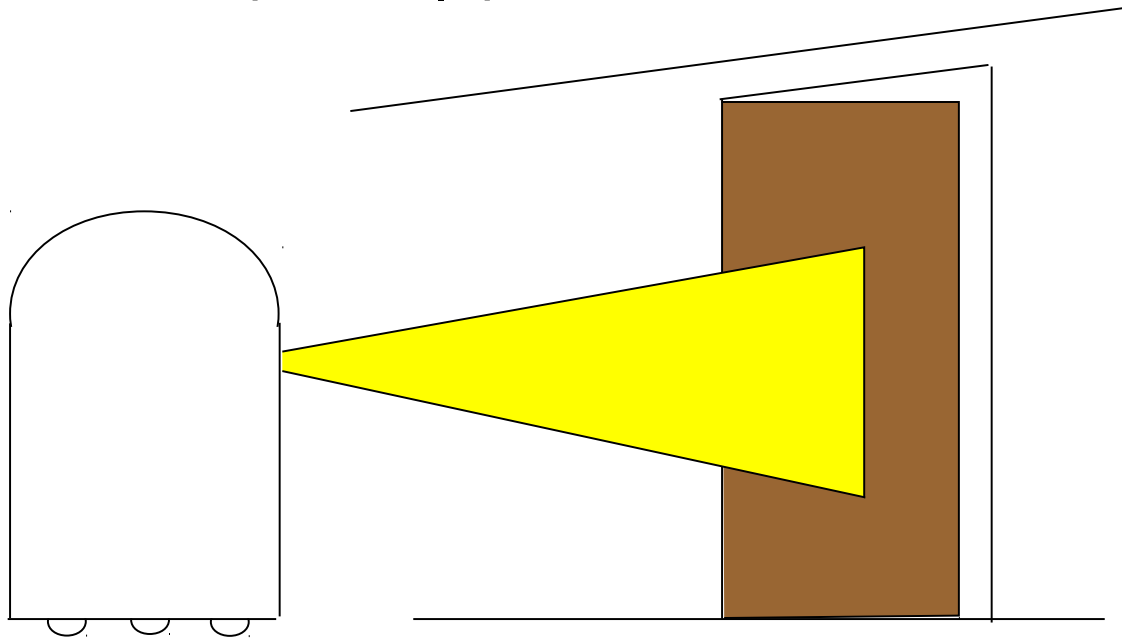
→ Exteroceptive sensors (e.g. laser, sonar, IR, GPS, camera, RFID, etc.)

A filtering approach is required to fuse all information

A Simple Example:

Estimating the state of a door

- Suppose a robot obtains measurement s
- What is $p(\text{Door}=\text{open}|\text{SensorMeasurement}=s)$?
- Short form: $p(\text{open}|s)$



Bayes Formula

$$p(A \wedge B) = p(A|B) p(B) = p(B|A) p(A)$$

\Rightarrow

$$p(A|B) = \frac{p(B|A) p(A)}{p(B)}$$

Causal vs. Diagnostic Reasoning

- We're interested in $p(open|s)$ (called *diagnostic* reasoning)
- Often *causal* knowledge like $p(s|open)$ is easier to obtain.
- From causal to diagnostic:

Apply Bayes rule:

$$p(open|s) = \frac{p(s|open) p(open)}{p(s)}$$

Normalization

$$p(\textit{open}|s) = \frac{p(s|\textit{open}) p(\textit{open})}{p(s)}$$

$$p(\neg \textit{open}|s) = \frac{p(s|\neg \textit{open}) p(\neg \textit{open})}{p(s)}$$

$$\begin{aligned} p(s) &= p(s \wedge \textit{open}) + p(s \wedge \neg \textit{open}) \\ &= p(s|\textit{open}) p(\textit{open}) + p(s|\neg \textit{open}) p(\neg \textit{open}) \end{aligned}$$

\Rightarrow

$$p(\textit{open}|s) = \frac{p(s|\textit{open}) p(\textit{open})}{p(s|\textit{open}) p(\textit{open}) + p(s|\neg \textit{open}) p(\neg \textit{open})}$$

Example

- $p(s|open) = 0.6$ $p(s|\neg open) = 0.3$
- $p(open) = p(\neg open) = 0.5$

$$p(open|s) = \frac{p(s|open) p(open)}{p(s|open) p(open) + p(s|\neg open) p(\neg open)}$$

$$p(open|s) = \frac{0.6 \cdot 0.5}{0.6 \cdot 0.5 + 0.3 \cdot 0.5} = \frac{2}{3} = 0.67$$

s raises the probability, that the door is open.

Integrating a second Measurement ...

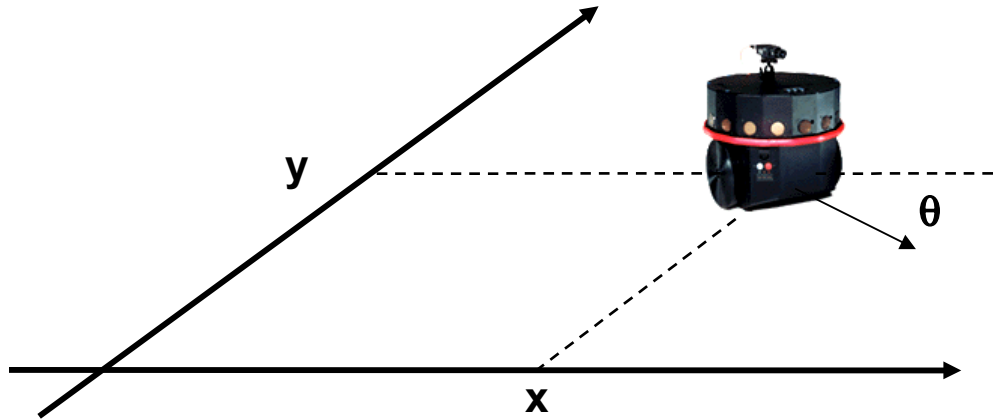
- New measurement s_2
- $p(s_2|open) = 0.5$ $p(s_2|\neg open) = 0.6$

$$p(open|s_2 s_1) = \frac{p(s_2|open) p(open|s_1)}{p(s_2|open) p(open|s_1) + p(s_2|\neg open) p(\neg open|s_1)}$$
$$p(open|s_2 s_1) = \frac{\frac{1}{2} \cdot \frac{2}{3}}{\frac{1}{2} \cdot \frac{2}{3} + \frac{3}{5} \cdot \frac{1}{3}} = \frac{5}{8} = 0.625$$

s_2 lowers the probability, that the door is open.

More formally, for the robot localization estimation problem:

$$x_r = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \quad \text{Robot pose}$$



$$x_{r,0:t} = \{x_{r,0}, x_{r,1}, \dots, x_{r,t}\} \quad \text{Robot poses from time 0 to time } t$$

$$z_{1:t} = \{z_1, z_2, \dots, z_t\} \quad \text{Robot exteroceptive measurements from time 1 to time } t$$

$$u_{0:t} = \{u_0, u_1, \dots, u_t\} \quad \text{Motion commands (or proprioceptive measurements) from time 0 to time } t$$

Markov Localization

- Let $l = \langle x, y, \theta \rangle$ represent a robot position in space
- $Bel(l)$ represents the robot's belief that it is at position l
 - $Bel(l)$ is a probability distribution, centered on the correct position, and updated based on two probabilistic models:

- Action (or motion) model:

$$Bel(l) \propto \int P(l | l', a) Bel(l') dl'$$

“Probability that an action a in position l moves the robot to position l , times the likelihood the robot is in position l , integrated over all possible ways robot could have reached position l ”

- Perception (or sensing) model:

$$Bel(l) \propto P(s | l) Bel(l)$$

“Probability that robot will perceive s , given that the robot is in position l , times the likelihood the robot is in position l ”

Markov Localization

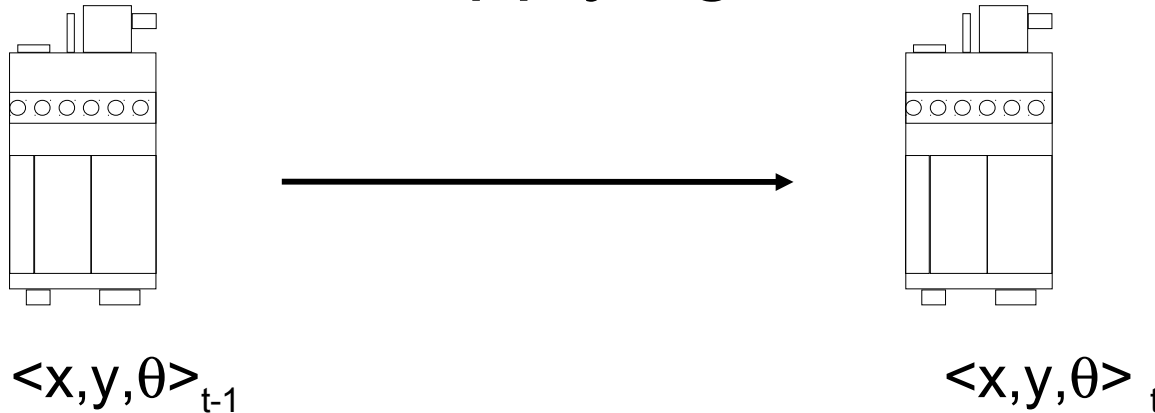
- “**Markov**” means the system obeys the “Markov Property”

$$Bel(l) \leftarrow Bel(l|a_T \dots a_2, a_1) = Bel(l|a_T)$$

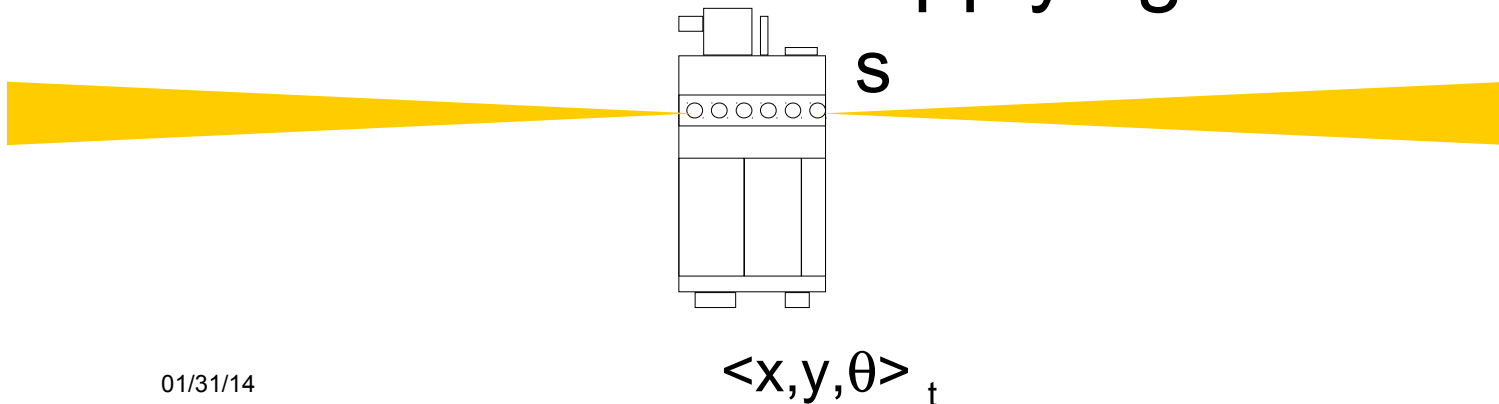
- “**Markov Property**”: the conditional probability of the future state is dependent only on the current state. It is independent of the past states.
- For the purposes of robot localization →
 - Future sensor readings are conditionally independent of past readings, given the true current position of the robot.
- Means we don't have to save all the prior sensor data and apply it each time we update beliefs on the robot's location.

Side Note: Common Terminology

Prediction Phase: Applying motion model



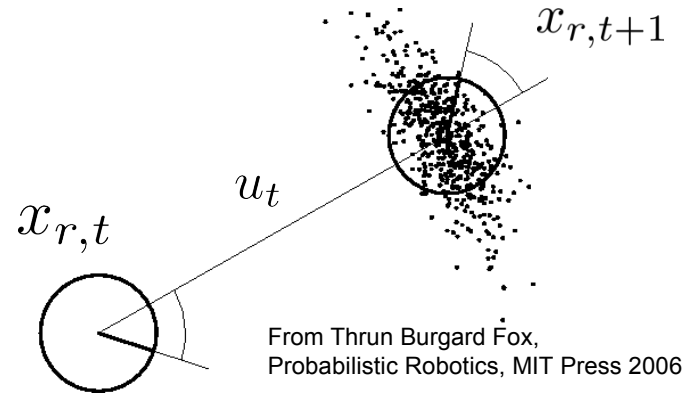
Measurement Phase: Applying sensor model



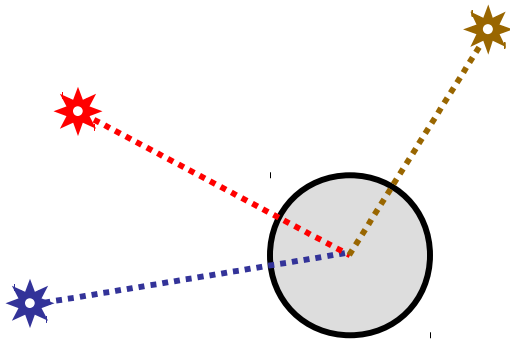
Robot description

- Motion model (proprioceptive sensors)

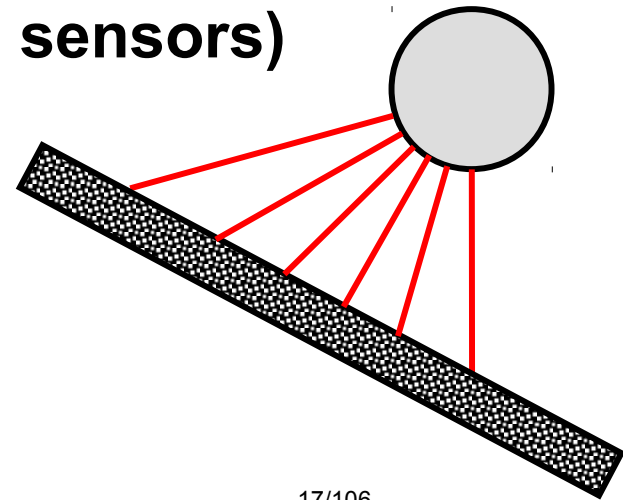
$$p(x_{r,t+1} | x_{r,t}, u_t)$$



- Measurement model (exteroceptive sensors)



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



Motion model (proprioceptive sensors)

$$p(x_{r,t+1} | x_{r,t}, u_t) = \text{prob}(x_{r,t+1} = x_r | x_{r,t}, u_t)$$

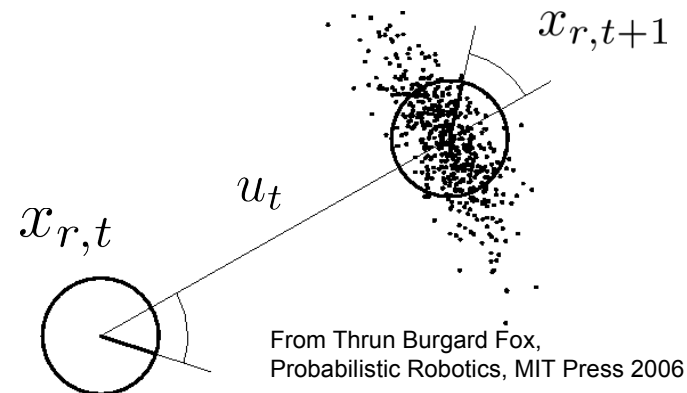
Assume:

- $x_{r,t}$ robot pose at time t
- u_t control input (i.e. motion command) at time t

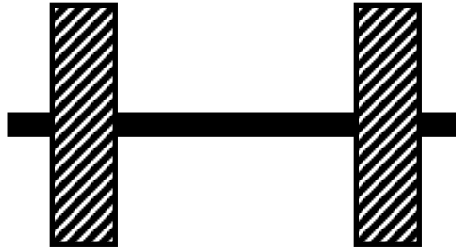
Which is the pose at time $t+1$?

Usually, u_t is obtained through proprioceptive sensors (e.g. encoders, gyroscopes, etc) → Odometry motion model

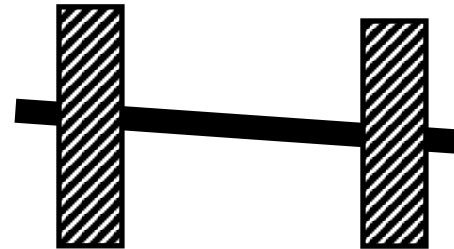
Due to sensor noise, $x_{r,t+1}$ is available only through a probability density function.



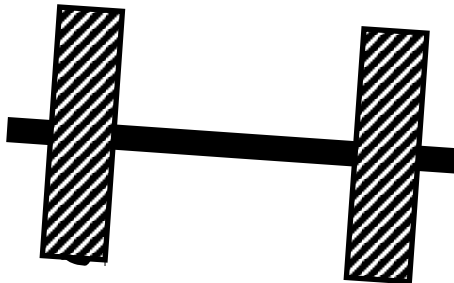
Possible sources of noise



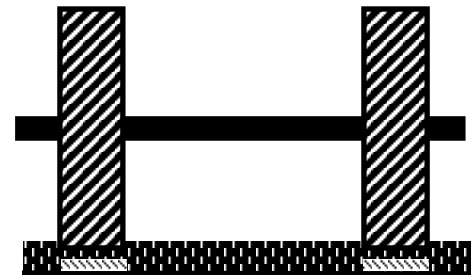
Ideal case



Different wheel
diameters (systematic source of noise)



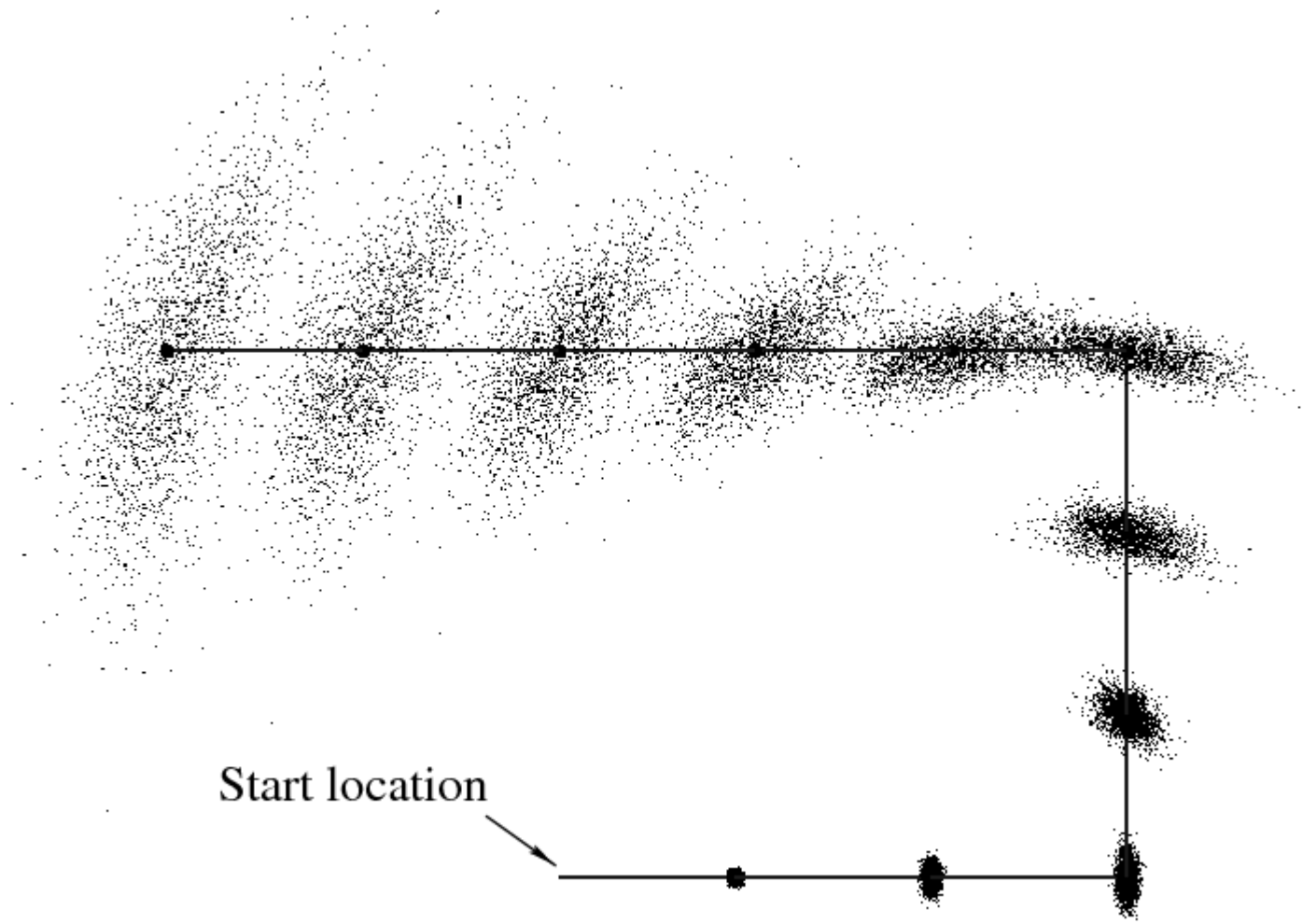
Bump (casual source of noise)



Carpet (casual source of noise)

and many more (e.g. distance between the wheels) ...

Accumulation of the pose estimation error under the robot motion
(only proprioceptive measurements)



Discretized unicycle model:

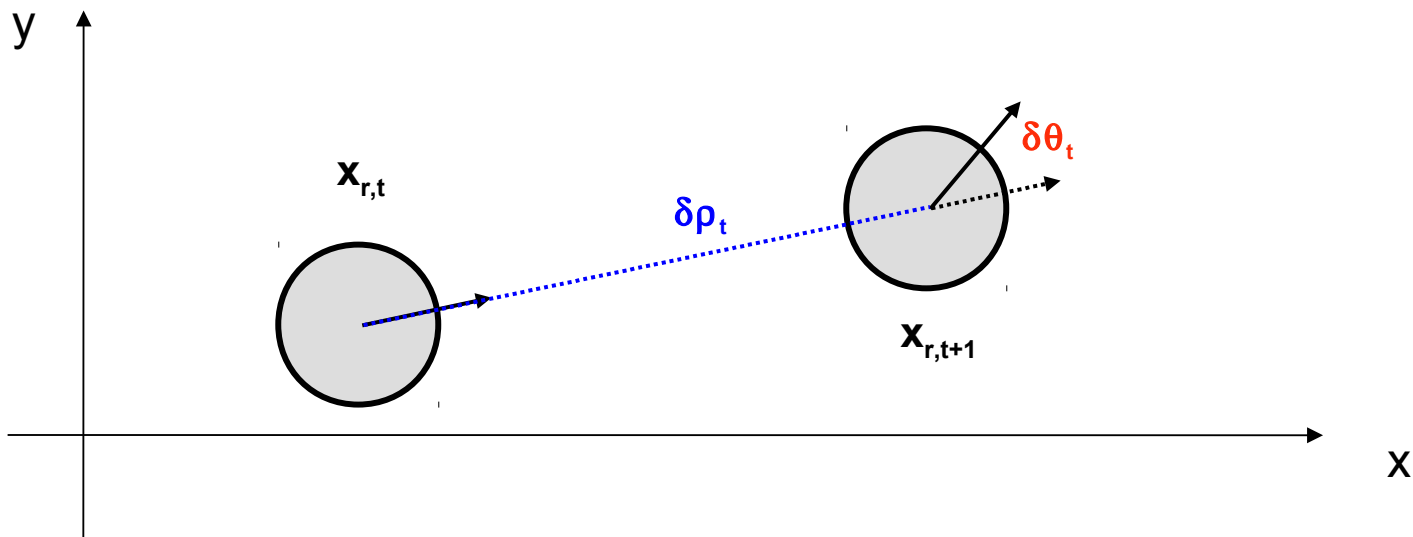
$$x_{t+1} = x_t + \boxed{\delta\rho_t} \cos(\theta_t)$$

$$y_{t+1} = y_t + \delta\rho_t \sin(\theta_t)$$

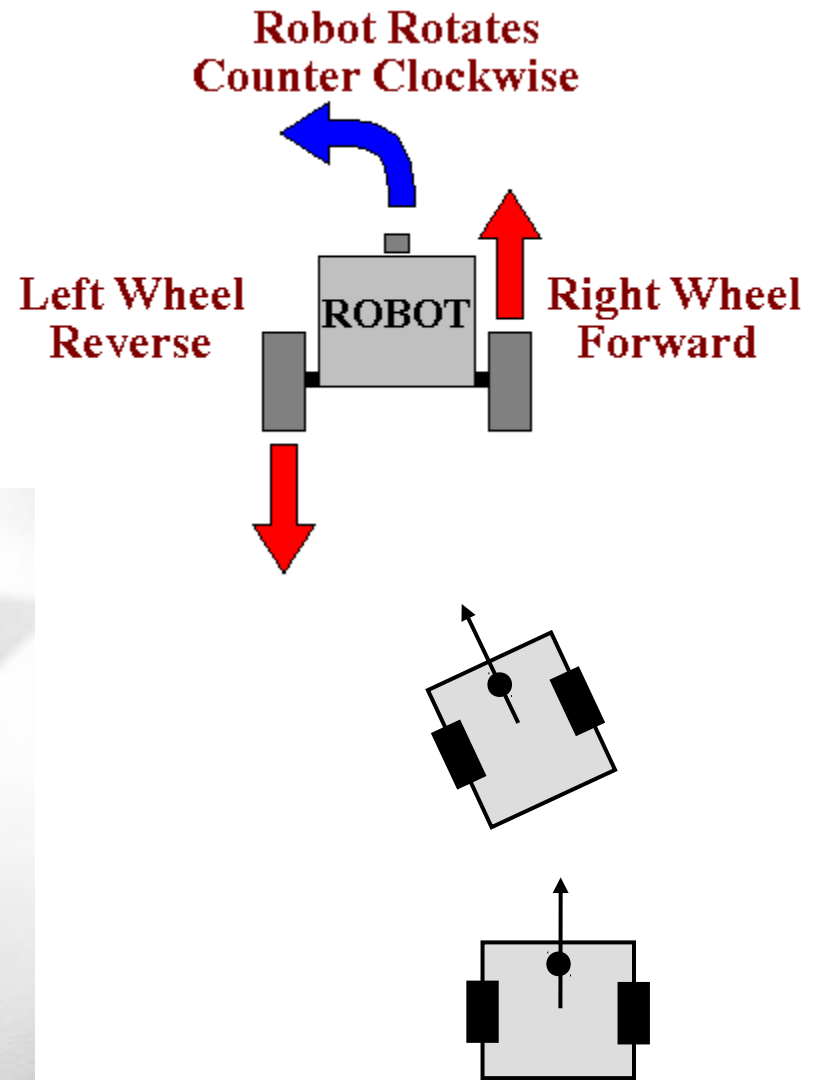
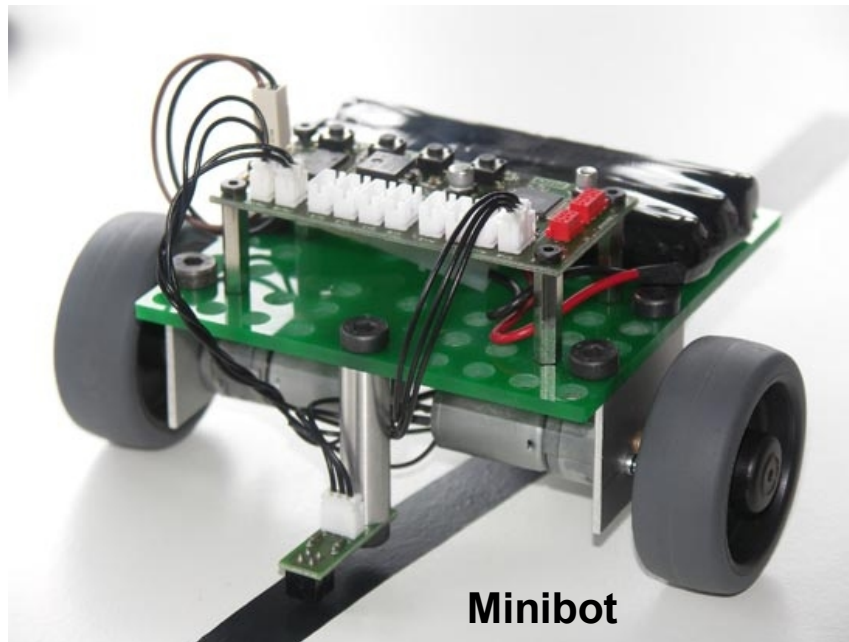
$$\theta_{t+1} = \theta_t + \boxed{\delta\theta_t}$$

Robot shift in time interval $[t, t+1]$

Robot rotation in time interval $[t, t+1]$



Discretized unicycle model: the differential drive case





$u_{R,t}$ = distance covered by the **Right wheel** in $[t, t+1]$

$u_{L,t}$ = distance covered by the **Left wheel** in $[t, t+1]$

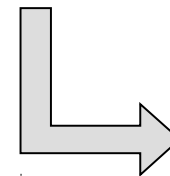
d = distance between the wheels

If the sampling time is small enough:

$$\delta \rho_t \approx D \delta \theta_t$$

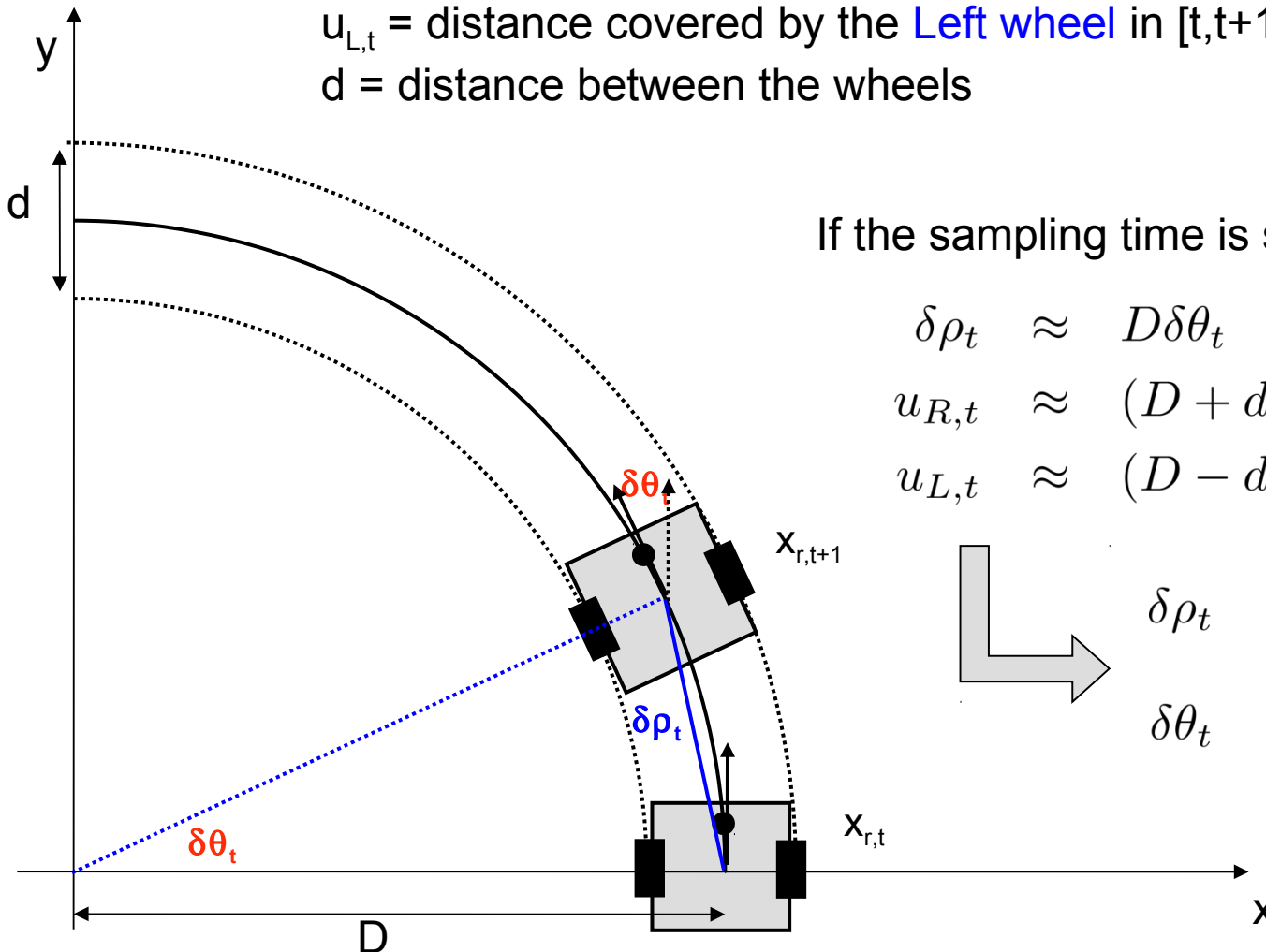
$$u_{R,t} \approx (D + d/2) \delta \theta_t$$

$$u_{L,t} \approx (D - d/2) \delta \theta_t$$



$$\delta \rho_t \approx \frac{u_{R,t} + u_{L,t}}{2}$$

$$\delta \theta_t \approx \frac{u_{R,t} - u_{L,t}}{d}$$



There is a **bijective relation** between $(\delta\rho, \delta\theta)$ and (u_R, u_L) :

$$\begin{aligned} \delta\rho_t &= \frac{u_{R,t} + u_{L,t}}{2} \\ \delta\theta_t &= \frac{u_{R,t} - u_{L,t}}{d} \end{aligned} \quad \longleftrightarrow \quad \begin{aligned} u_{R,t} &= \delta\rho_t + \frac{d}{2} \delta\theta_t \\ u_{L,t} &= \delta\rho_t - \frac{d}{2} \delta\theta_t \end{aligned}$$

Mismatch between **real** $(u_{R,t}, u_{L,t})$ and **measured** $(u_{R,t}^e, u_{L,t}^e)$ distances covered by the wheels of the robot. The encoder readings are related to the real distances covered by the two wheels as follows:

$$u_{R,t}^e = u_{R,t} + n_{R,t}$$

$$u_{L,t}^e = u_{L,t} + n_{L,t}$$

(**casual** source of noise)

$$n_{R,t} \sim \mathcal{N}(0, K_R |u_{R,t}|)$$

$$n_{L,t} \sim \mathcal{N}(0, K_L |u_{L,t}|)$$

Given:

- $\mathbf{x}_{r,t} = [x, y, \theta]^T$ robot pose at time t ,
- encoder reading $\mathbf{u}_t = [u_R^e, u_L^e]^T$ (i.e. result of the control input) relative to the interval $[t, t+1]$,

we want to estimate the probability that the robot pose at time $t+1$ will be a given $\mathbf{x}_{r,t+1} = [x', y', \theta']^T$ (under the differential drive model considered), i.e.:

$$p(x_{r,t+1} | x_{r,t}, u_t) = p([x', y', \theta']^T | [x, y, \theta]^T, (u_R^e, u_L^e))$$

We proceed as follows:

Actual robot displacement if we want to arrive at $[x', y', \theta']^T$ from $[x, y, \theta]^T$:

$$\begin{aligned}\delta\rho &= \sqrt{(x - x')^2 + (y - y')^2} \\ \delta\theta &= \theta' - \theta\end{aligned}$$

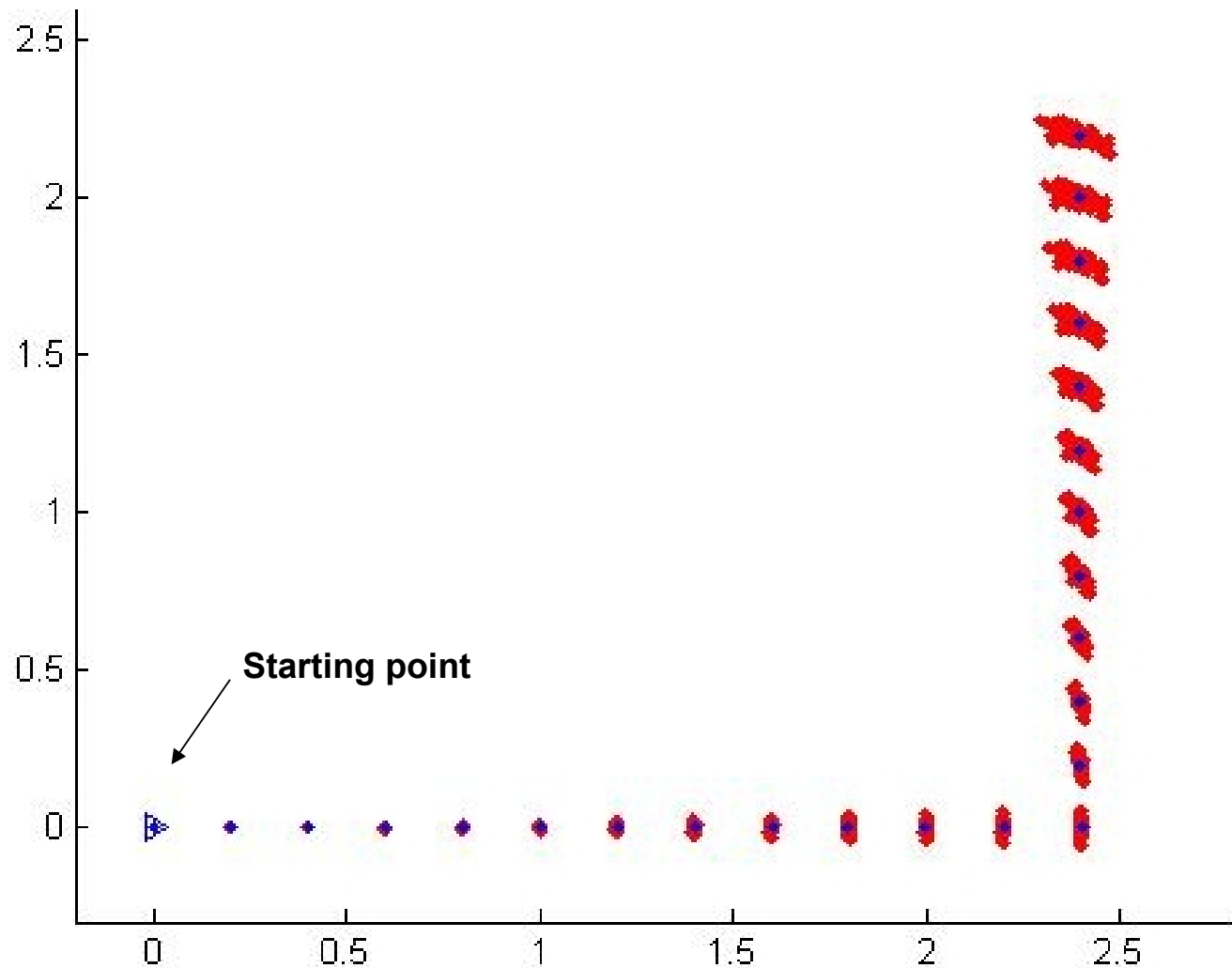
Corresponding distances covered by the two wheels, while the actual encoder readings are (u_R^e, u_L^e) :

$$\begin{aligned}u_R &= \delta\rho + \frac{d}{2} \delta\theta \\ u_L &= \delta\rho - \frac{d}{2} \delta\theta\end{aligned}$$

The probability of arriving in $[x', y', \theta']^T$ from $[x, y, \theta]^T$ is the **probability of reading (u_R^e, u_L^e) when the actual wheel shifts are (u_R, u_L)**

$$\begin{aligned}p([x', y', \theta']^T | [x, y, \theta]^T, (u_R^e, u_L^e)) \\ = \text{prob}(n_{R,t} = u_R - u_R^e) \cdot \text{prob}(n_{L,t} = u_L - u_L^e)\end{aligned}$$

Estimation (red points) of the path covered by a differential drive robot.
Blue = real position in each step.



Measurement model (exteroceptive sensors)

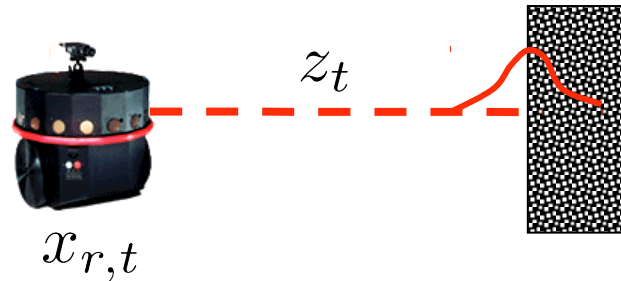
$$p(z_t | x_{r,t})$$

Assume:

- $x_{r,t}$ robot pose at time t

Which is the reading z_t of a sensor at time t ?

Due to sensor noise, z_t is available only through a probability density function.

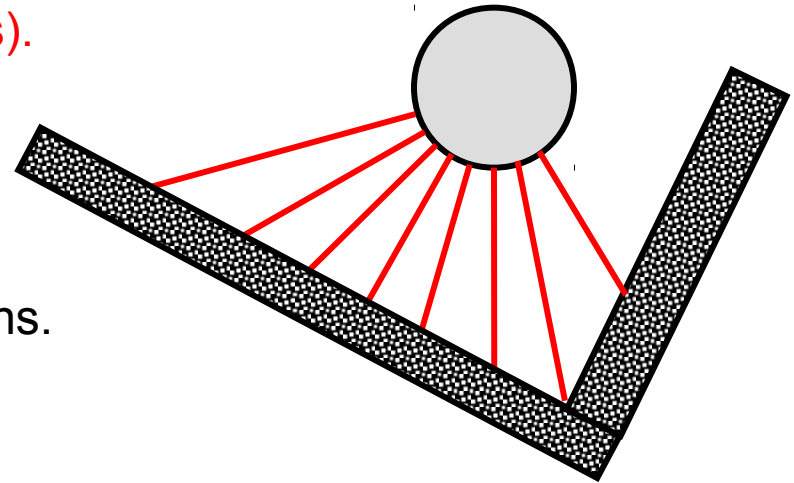


Measurement model (exteroceptive sensors)

Two typical settings are considered:

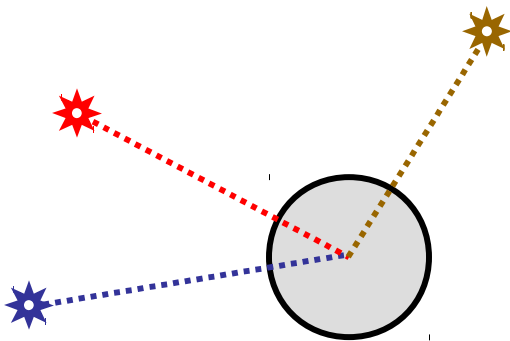
Range Finder (e.g. laser or sonar readings).

The environment is known and is represented by line segments (e.g. the walls). The robot senses the distance from them at some given directions.



Landmark measurements.

The environment is characterized by features (artificial or natural landmarks), whose position is known and whose identity can be known (e.g. RFID tags) or must be estimated (data association problem). The robot measures the distance and/or the bearing to these fixed points.



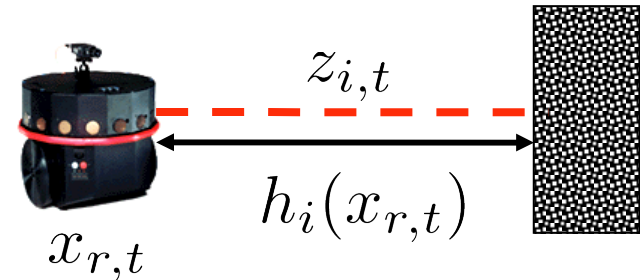
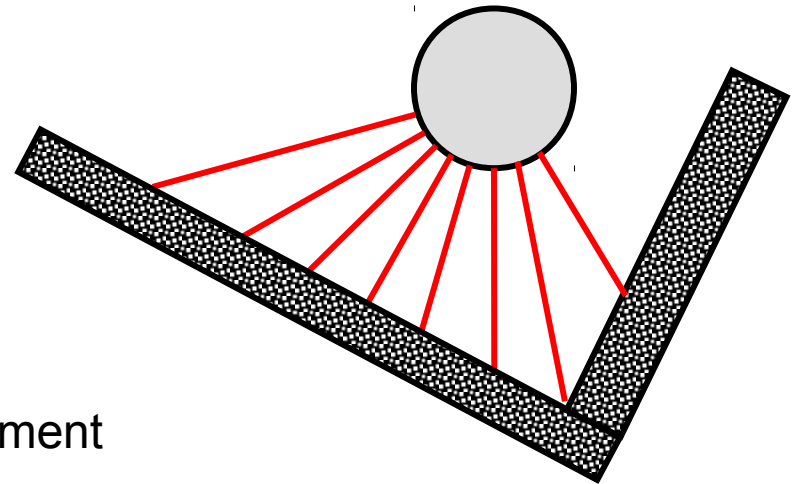
Measurement model: the Range Finder case

$$z_t = [z_{1,t}, z_{2,t}, \dots, z_{m,t}]^T$$

Each measurement $z_{i,t}$ is a range measurement typically based on the time of flight of the beam (e.g. of a sonar or a laser) generated by sensor i and reflected by an obstacle

$z_{i,t}$ Measurement given by beam i

$h_i(x_{r,t})$ Distance of sensor i from the reflecting obstacle (ideal reading of the sensor)



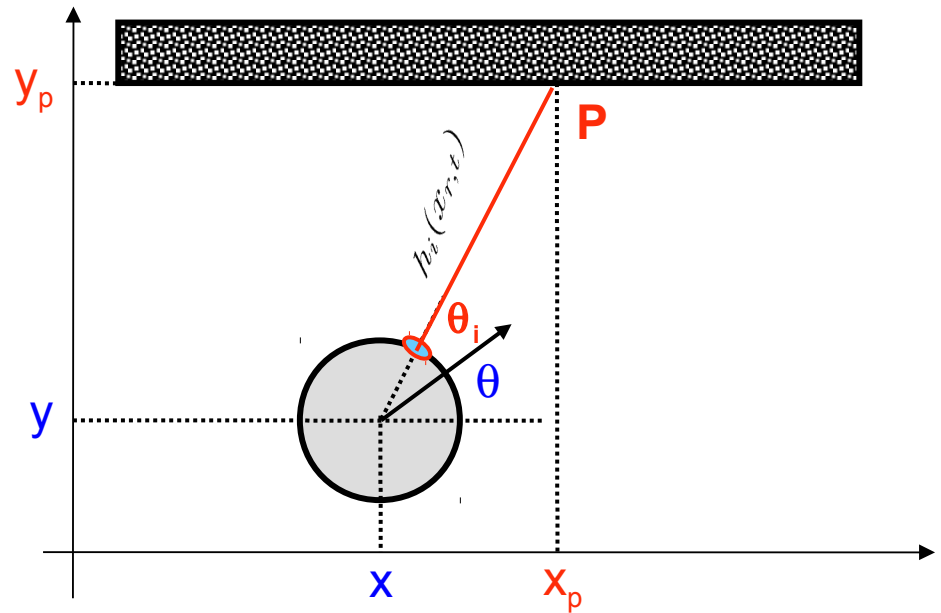
$$z_{i,t} = h_i(x_{r,t}) + \nu_{i,t}$$
$$\nu_{i,t} \sim \mathcal{N}(0, \zeta_i^2)$$

Given:

- $\mathbf{x}_{r,t} = [x, y, \theta]^T$ robot pose at time t ,

we want to estimate the probability that the measurement of sensor i (with orientation θ_i with respect to the robot motion direction) at time t is a given z_t .

- First compute the distance $h_i(\mathbf{x}_{r,t})$ between sensor i and the reflecting point **P**
(this is possible given the robot pose since the environment is assumed known)
- Then:



$$p(z_t | x_{r,t}) = \text{prob}(\nu_t = z_t - h_i(x_{r,t}))$$

Measurement model:

landmark measurements

Let $\mathbf{x}_{r,t} = [x, y, \theta]^T$ be the robot pose at time t

Assuming known the identities (and the position) of the landmarks, for each landmark $\mathbf{L}_i = (x_i, y_i)$ there is a measurement

$$z_{i,t} = \left(r_{i,t} + \nu_{i,t}^r, \phi_{i,t} + \nu_{i,t}^\phi \right)$$

where

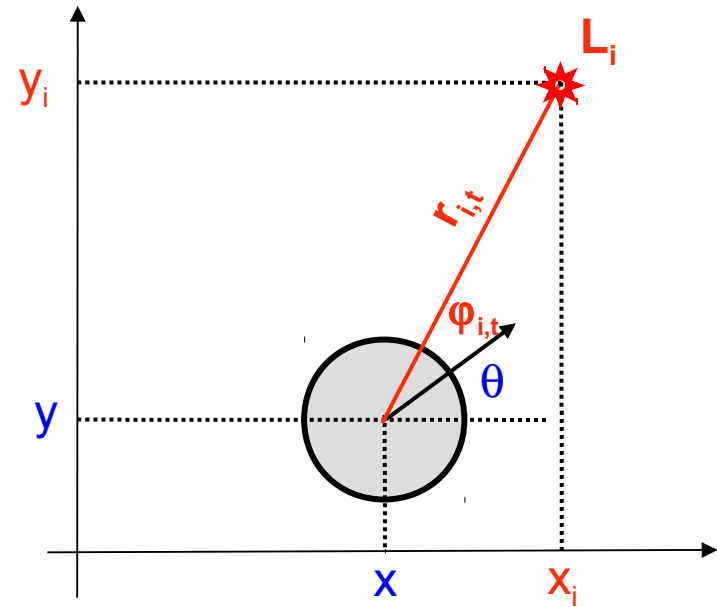
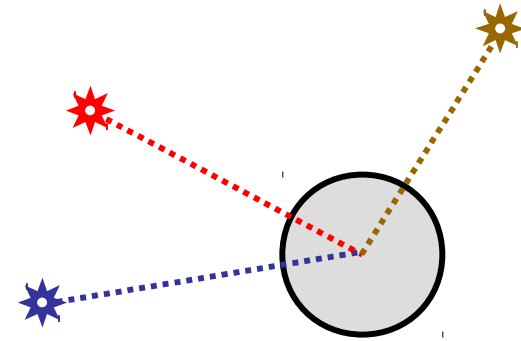
$$r_{i,t} = \sqrt{(x_i - x)^2 + (y_i - y)^2}$$

$$\phi_{i,t} = \text{atan2}((y_i - y), (x_i - x)) - \theta$$

and

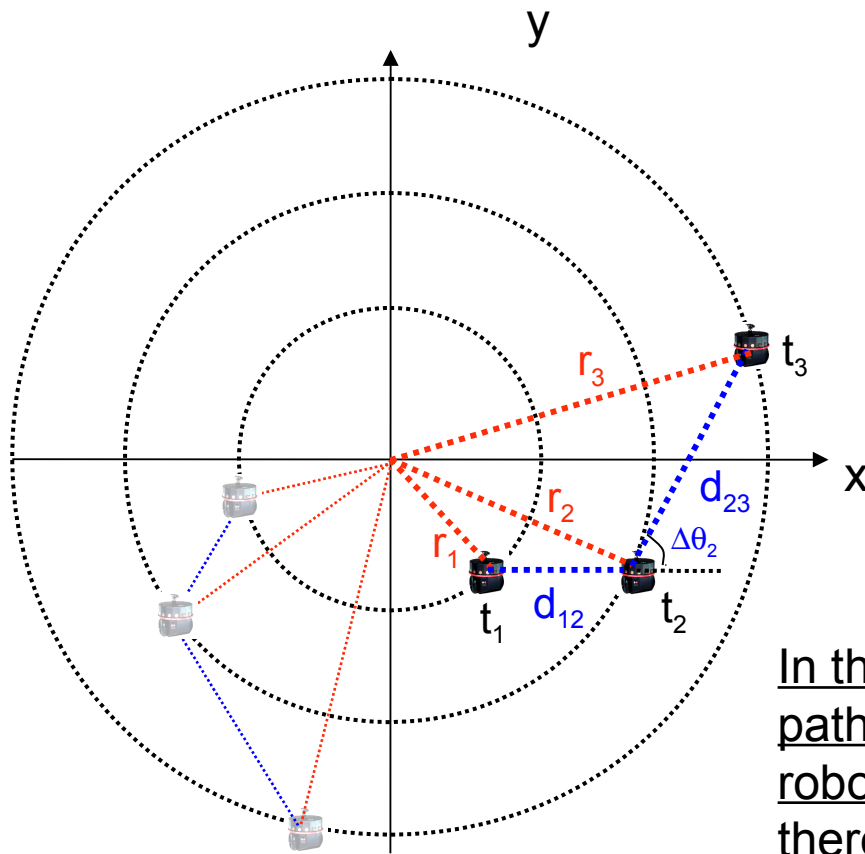
$$\nu_{i,t}^r \sim \mathcal{N}(0, \zeta_{i,r}^2)$$

$$\nu_{i,t}^\phi \sim \mathcal{N}(0, \zeta_{i,\phi}^2)$$



Observability issue

The robot may have not enough sensors to determine its pose, even if assuming perfect (i.e. without noise) measurements.



The robot is equipped with encoder readings and can measure the distance r to a landmark located in the origin.

r_i = range measurement at time t_i

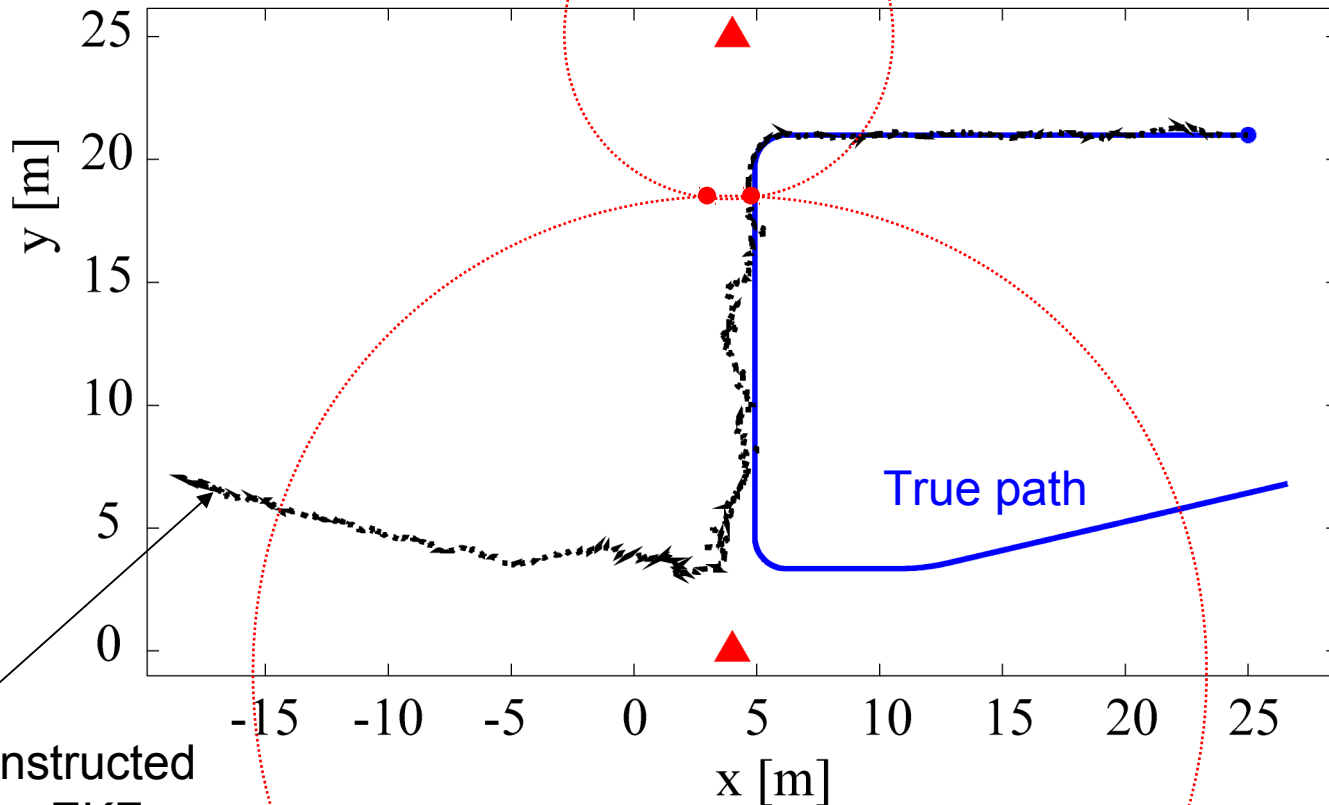
d_{ij} = distance covered
(and measured) by the
robot in time interval $[t_i, t_j]$

$\Delta\theta_2$ = robot turn at time t_2 (known
from odometry)

In the figure we represent two different robot paths agreeing with measurements. The robot pose **is not completely observable**: there is a radial symmetry around the origin

Case the robot measures (with noise) the distance from two landmarks (in addition to the odometry). The robot pose **is not completely observable**.

Observability issue



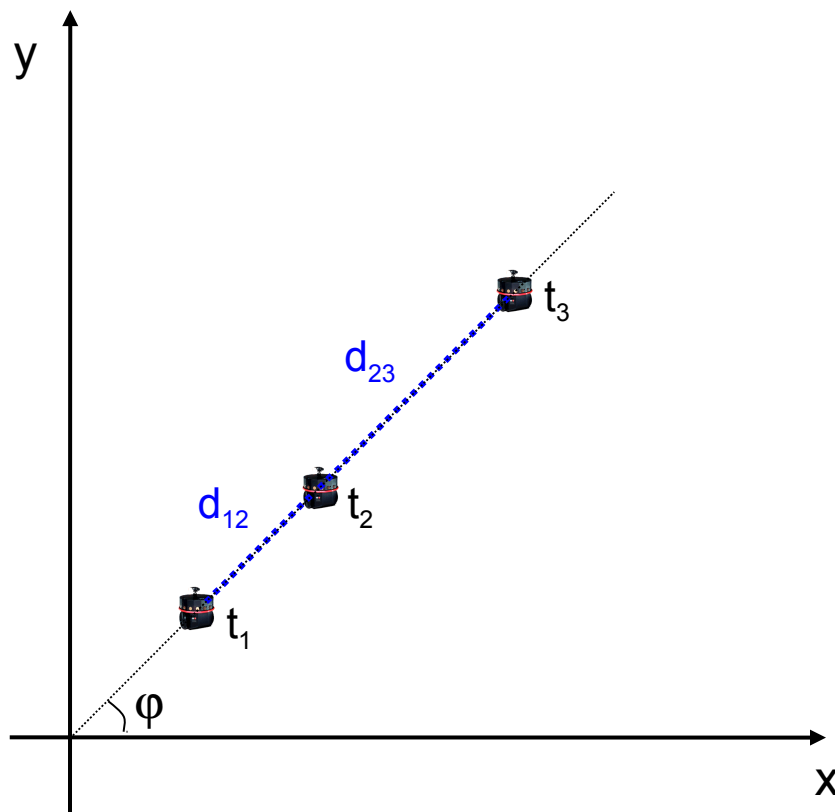
Path reconstructed
through the EKF
(position tracking)

The noise makes the estimated pose jump to the specular path.

In the nonlinear case, the observability depends on the particular control input considered.

Hence, the possibility of localizing a mobile robot may depend on the particular motion executed which in any case affects the localization performance

→ Active Localization

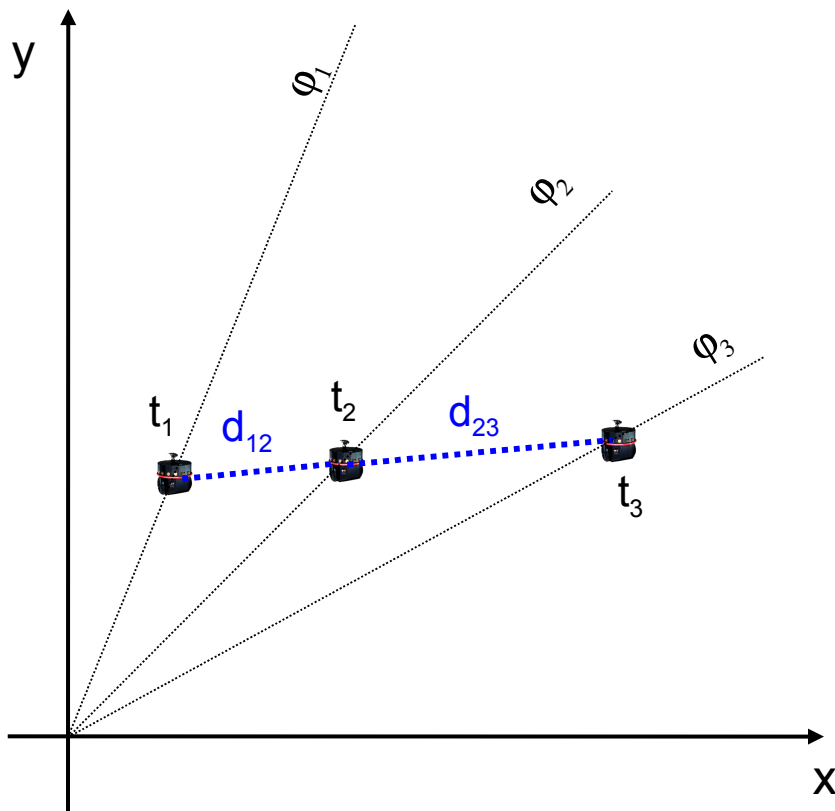


The robot is equipped with odometry and measures the angle φ (a non symmetric landmark is located in the origin).

d_{ij} = distance covered (and measured)
by the robot in time interval $[t_i, t_j]$

Under this motion (radial) the distance of the robot from the origin (hence x and y) **cannot be determined** but only the orientation $\theta = \varphi$

A different kind of motion would make the pose observable (active localization):



The robot is equipped with odometry and measures the angle ϕ (a non symmetric landmark is located in the origin).

d_{ij} = distance covered (and measured) by the robot in time interval $[t_i, t_j]$

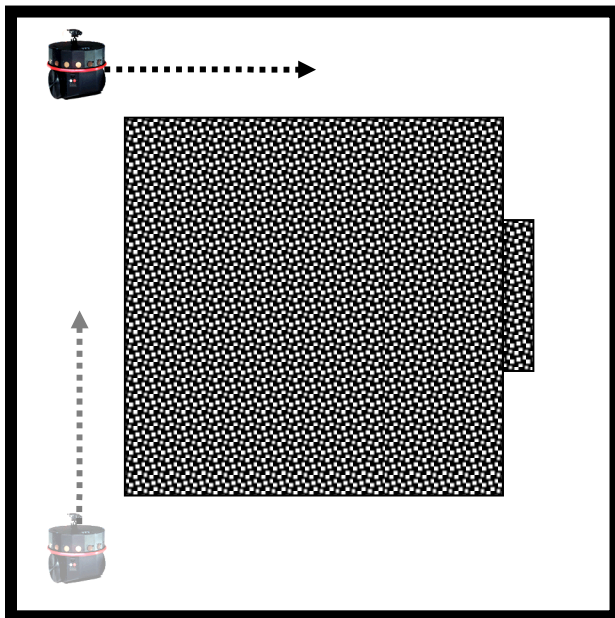
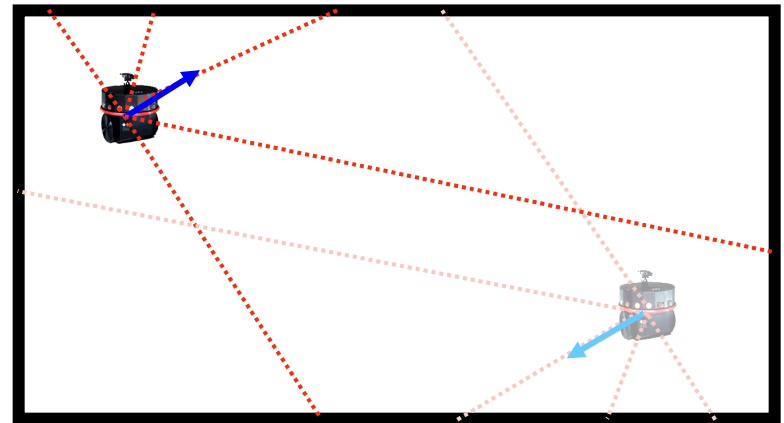
Under this motion all the pose of the robot **can be determined**

Often localization is performed under a generic motion (passive localization): the robot tries to localize itself while performing its mission (once a satisfactory pose estimation has been reached) or under a random motion away from obstacles (at the beginning, when the pose is still completely unknown).

This is because usually mobile robots are equipped with several sensors: they can often localize themselves even if not moving (e.g. if measuring the distance to at least 3 non collinear landmarks).

Nevertheless...

problems may arise with **symmetrical** environments (the robot has odometry + rangefinder, e.g. sonar or laser).



A proper robot motion may disambiguate in some cases the pose estimation.

The Bayes filter is **not a practical algorithm**

→ **Gaussian Filters**: Extended Kalman Filter (EKF),
Unscented Kalman Filter (UKF),
Extended Information Filter (EIF)

→ **Non Parametric Filters**: Histogram Filter (HF),
Particle Filter (PF)

Grid representation of the belief:

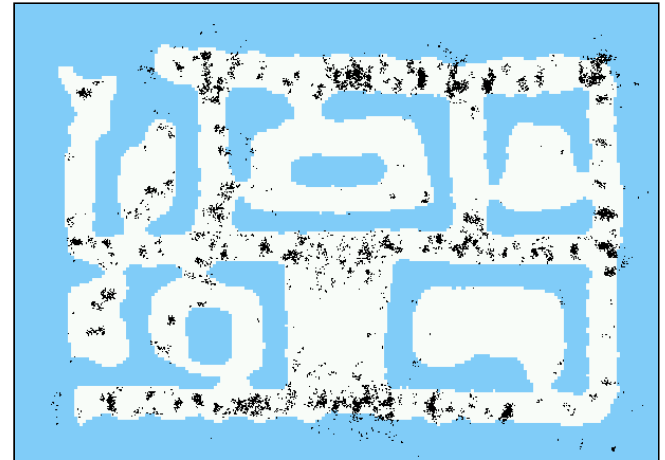
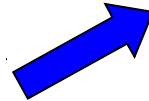
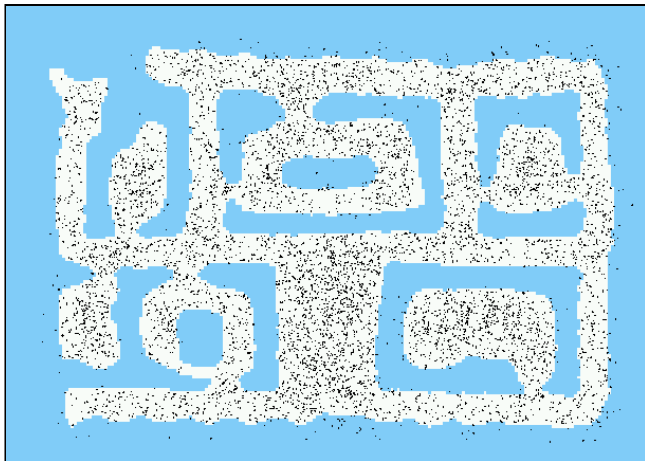
$$bel_t(x_r) \quad := \quad \{p_t(c_k)\}_{c_k \in \Omega_D}$$



Particle filter

Particle representation of the belief:

$$bel_t(x_r) = \{x_{r,t}^m, w_t^m\}_{m=1,\dots,M}$$



From Thrun Burgard Fox, Probabilistic Robotics, MIT Press 2006

MCL with Particle Filter: Setting up the problem

The robot does (or can be modeled to) alternate between

- sensing -- getting range observations $z_1, z_2, \dots, z_{t-1}, z_t$
 - acting -- driving around (or ferrying?) u_1, u_2, \dots, u_{t-1}
-

We want to know x_t -- the position of the robot at time t

- but we'll settle for $p(x_t)$ -- the probability distribution for x_t



What *kind* of thing is $p(x_t)$?

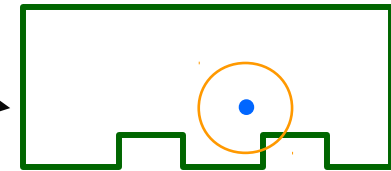
We do know m (the map of the environment)

- the sensor model $p(z \mid x, m)$
- the accuracy of desired action u $p(x_{\text{new}} \mid x_{\text{old}}, u, m)$

Robot modeling

$p(z \mid x, m)$ sensor model

map m and location r



$$p(\text{shaded region} \mid x, m) = .75$$

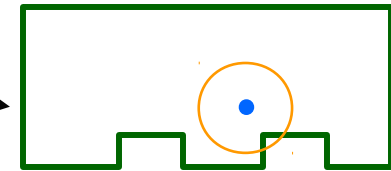
$$p(\text{shaded region} \mid x, m) = .05$$

potential observations Z

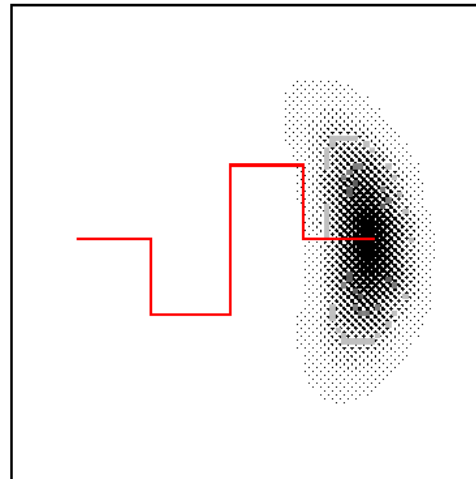
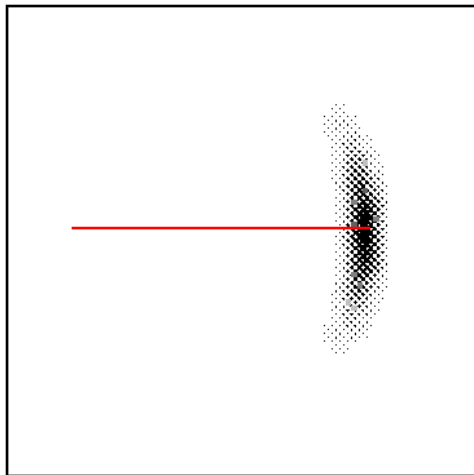
Robot modeling

$p(o \mid r, m)$ sensor model

map m and location r



$p(x_{\text{new}} \mid x_{\text{old}}, u, m)$ action model



$$p(\text{circle with blue dot and black sector} \mid r, m) = .75$$

$$p(\text{circle with blue dot and black sector} \mid r, m) = .05$$

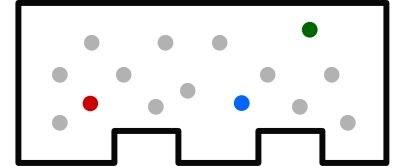
potential observations O

“probabilistic kinematics” -- encoder uncertainty

- red lines indicate commanded action
- the cloud indicates the likelihood of various final states

Monte Carlo Localization

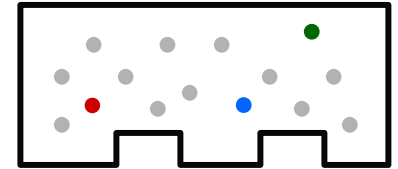
1) Start by assuming $p(x_0)$ is the uniform distribution.
Take K samples of x_0 and weight each with an
importance factor of $1/K$




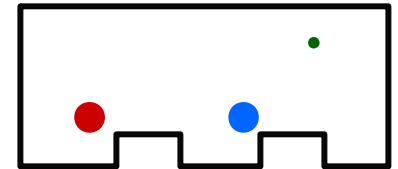
Monte Carlo Localization

1) Start by assuming $p(x_0)$ is the uniform distribution.

Take K samples of x_0 and weight each with an *importance factor* of $1/K$



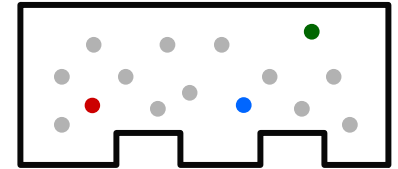
2) Get the current sensor observation, z_1 
For each sample point x_0 multiply the importance factor by $p(z_1 | x_0, m)$, and normalize
(make sure the importance factors add to 1)




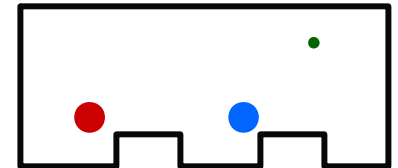
Monte Carlo Localization

1) Start by assuming $p(x_0)$ is the uniform distribution.

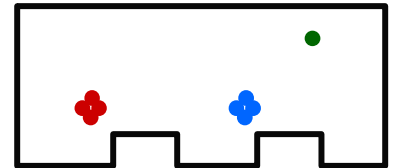
Take K samples of x_0 and weight each with an *importance factor* of $1/K$



2) Get the current sensor observation, z_1 
For each sample point x_0 multiply the importance factor by $p(z_1 | x_0, m)$, and normalize
(make sure the importance factors add to 1)



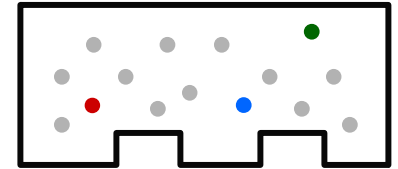
3) Create x_1 samples by dividing up large clumps
Each point spawns new ones in proportion to its importance factor




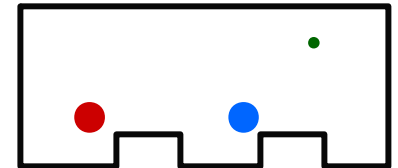
Monte Carlo Localization

1) Start by assuming $p(x_0)$ is the uniform distribution.

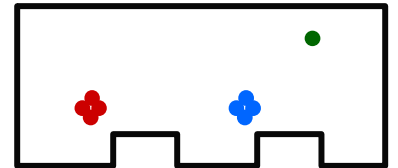
Take K samples of x_0 and weight each with an *importance factor* of $1/K$



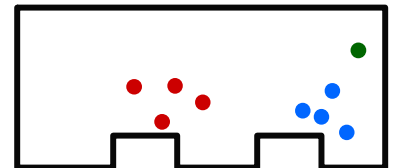
2) Get the current sensor observation, z_1 
For each sample point x_0 multiply the importance factor by $p(z_1 | x_0, m)$, and normalize
(make sure the importance factors add to 1)



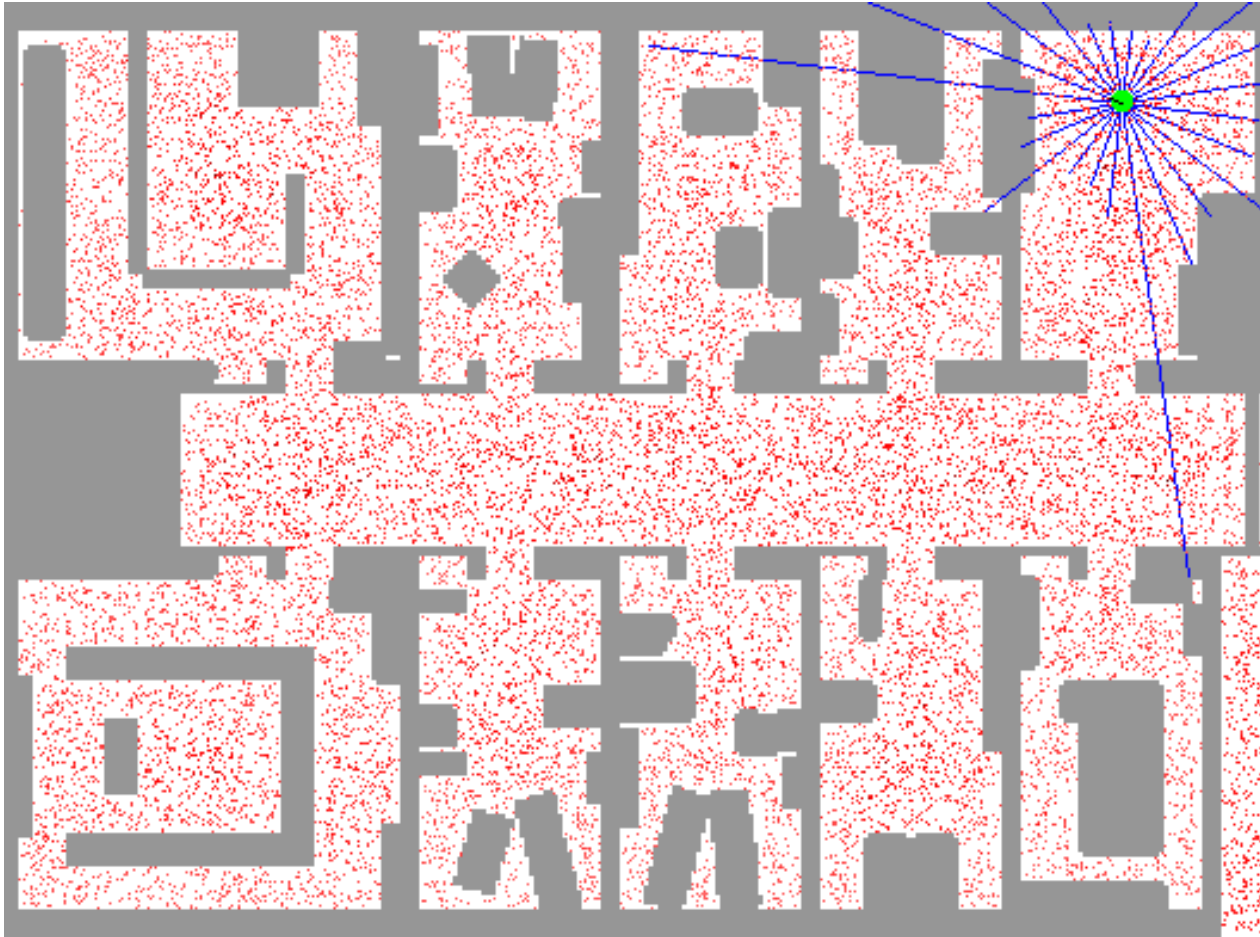
3) Create x_1 samples by dividing up large clumps
Each point spawns new ones in proportion to its importance factor



4) The robot moves, u_1
Each point spawns new ones in proportion to its importance factor



MCL in action



References

1. [S. Thrun, W. Burgard and D. Fox. "Probabilistic Robotics"](#). MIT Press, 2006
(and references therein)

Robot motion models (Ch. 5), measurement models (Ch. 6), Localization (Ch. 7-8), Filtering (Ch. 2-4), Active localization (Ch. 17).