Mobile Robots Lab
Localization using magnets

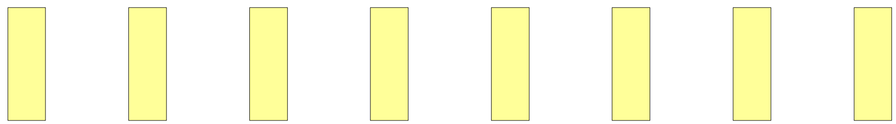
Presentation

The robot and the magnet sensor

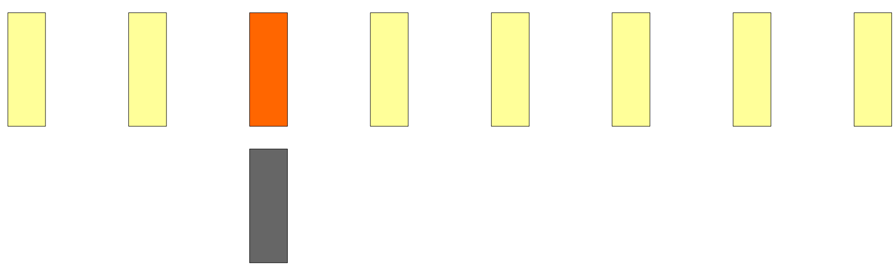
The reed switch

A normally open reed switch.
In a sufficiently intense magnetic field, the switch closes.

The magnet sensor

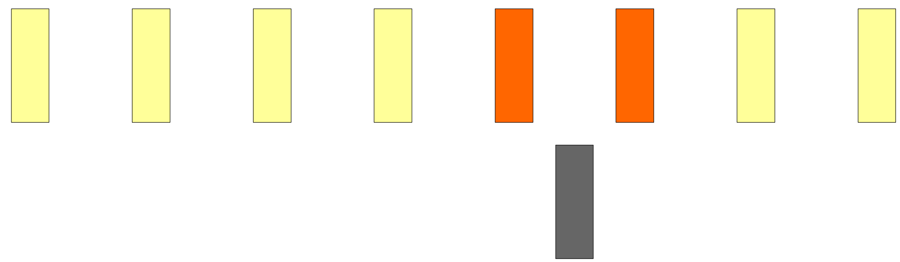


No magnet in the vicinity of the reed switches: all are open

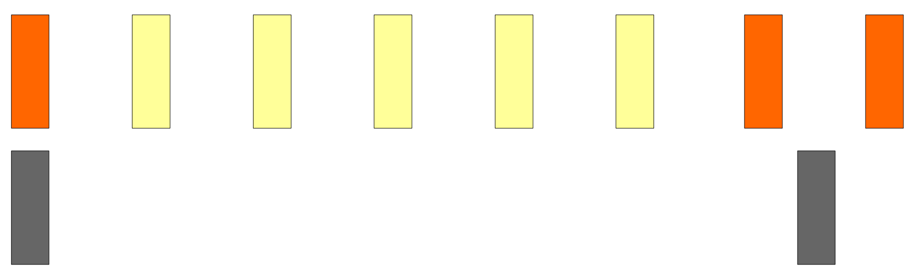


A magnet is right under reed switch 3, which is closed.

The magnet sensor

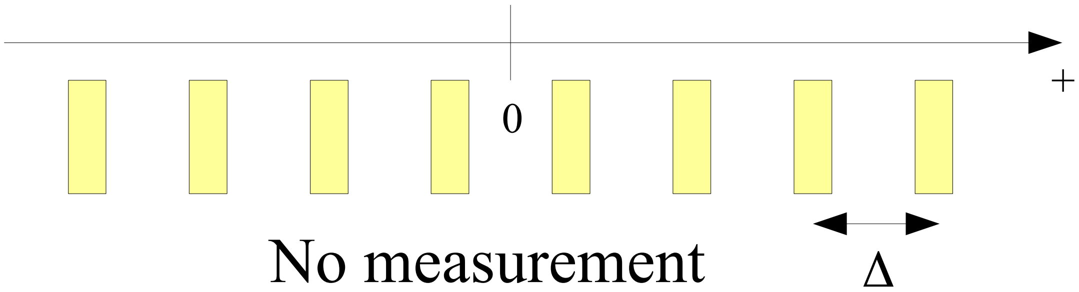


A magnet is right under switches 5 and 6, which are closed.

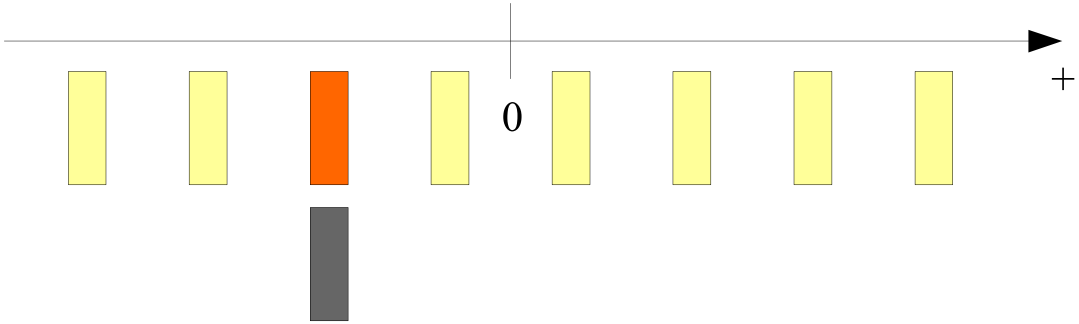


A magnet is right under switche 1, another under 7 and 8

The magnet sensor measurements

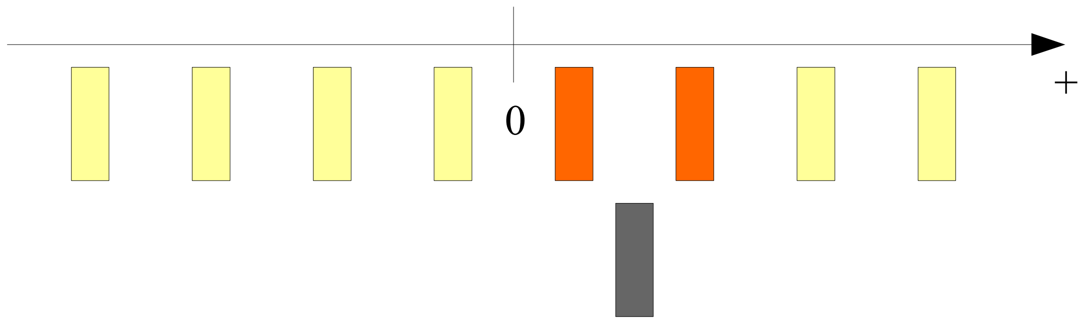


No measurement

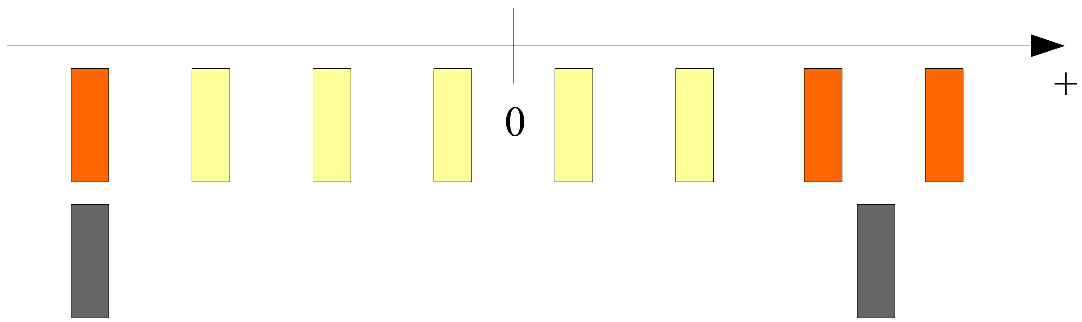


Measurement is -1.5 Δ

The magnet sensor measurements

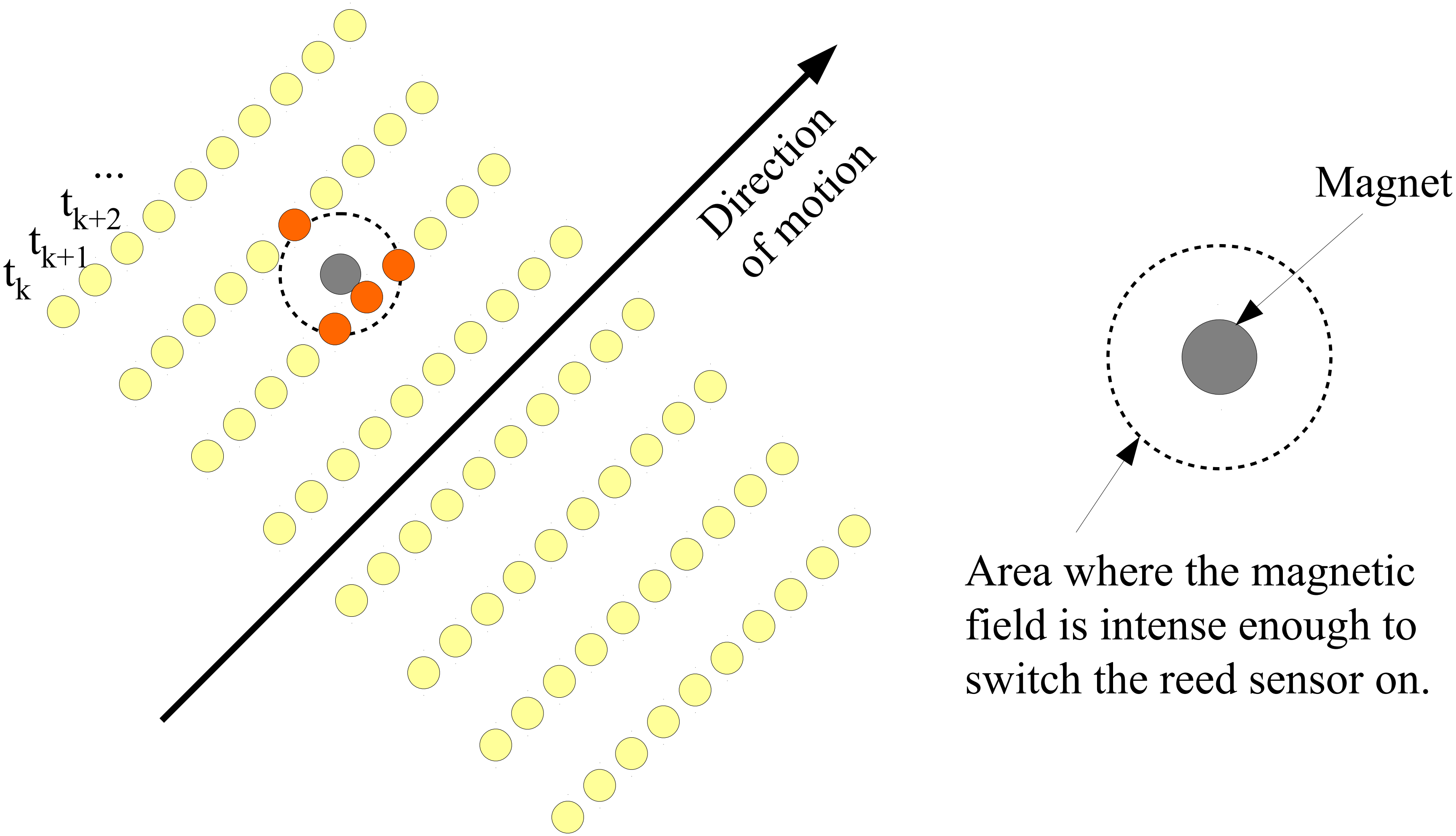


Measurement is $+\Delta$

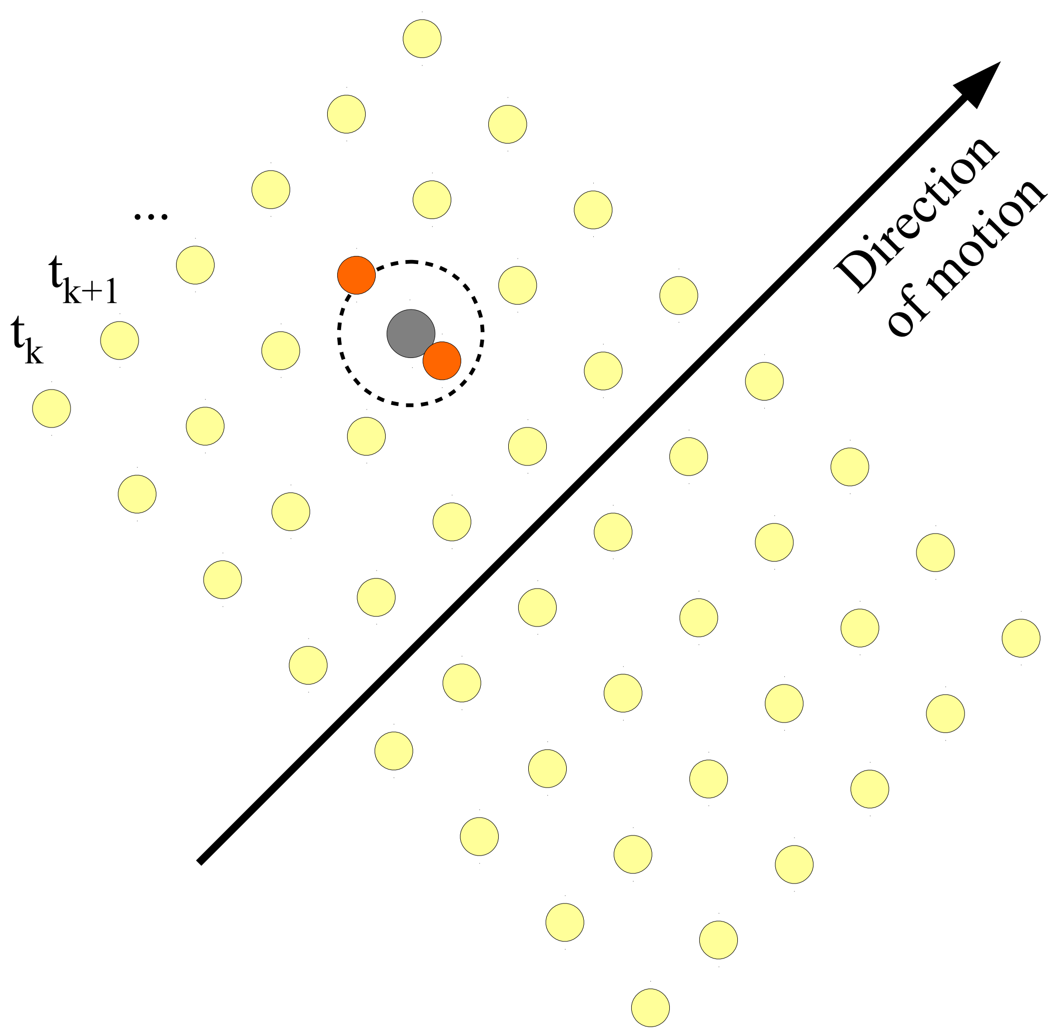


Two measurements: -3.5Δ and $+3\Delta$

Sensor passing over a magnet



Sensor passing over a magnet
(lower sampling rate)

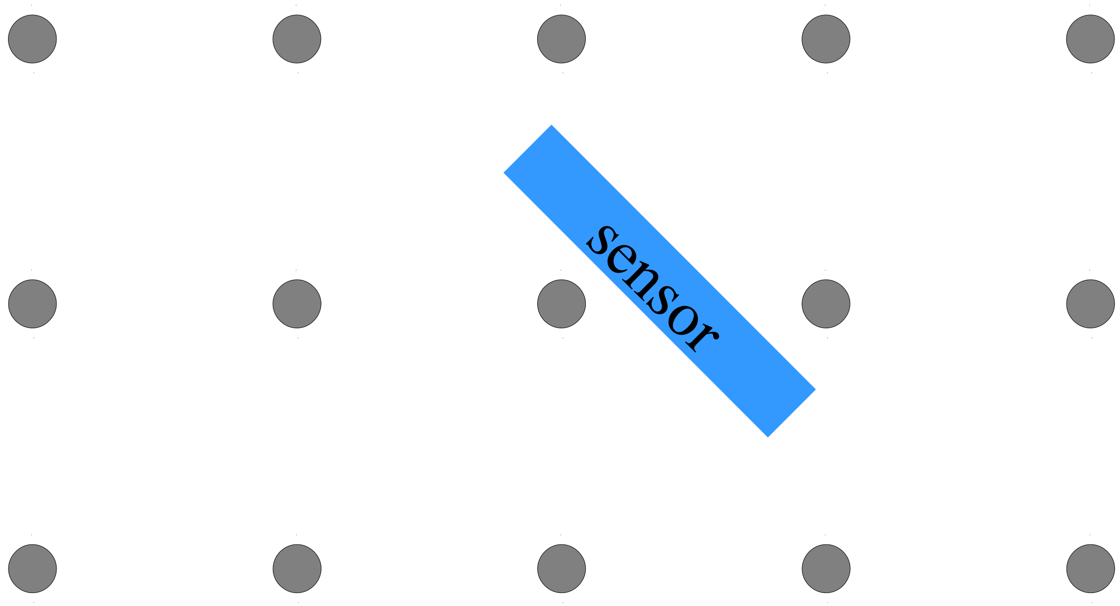


About the magnet sensor

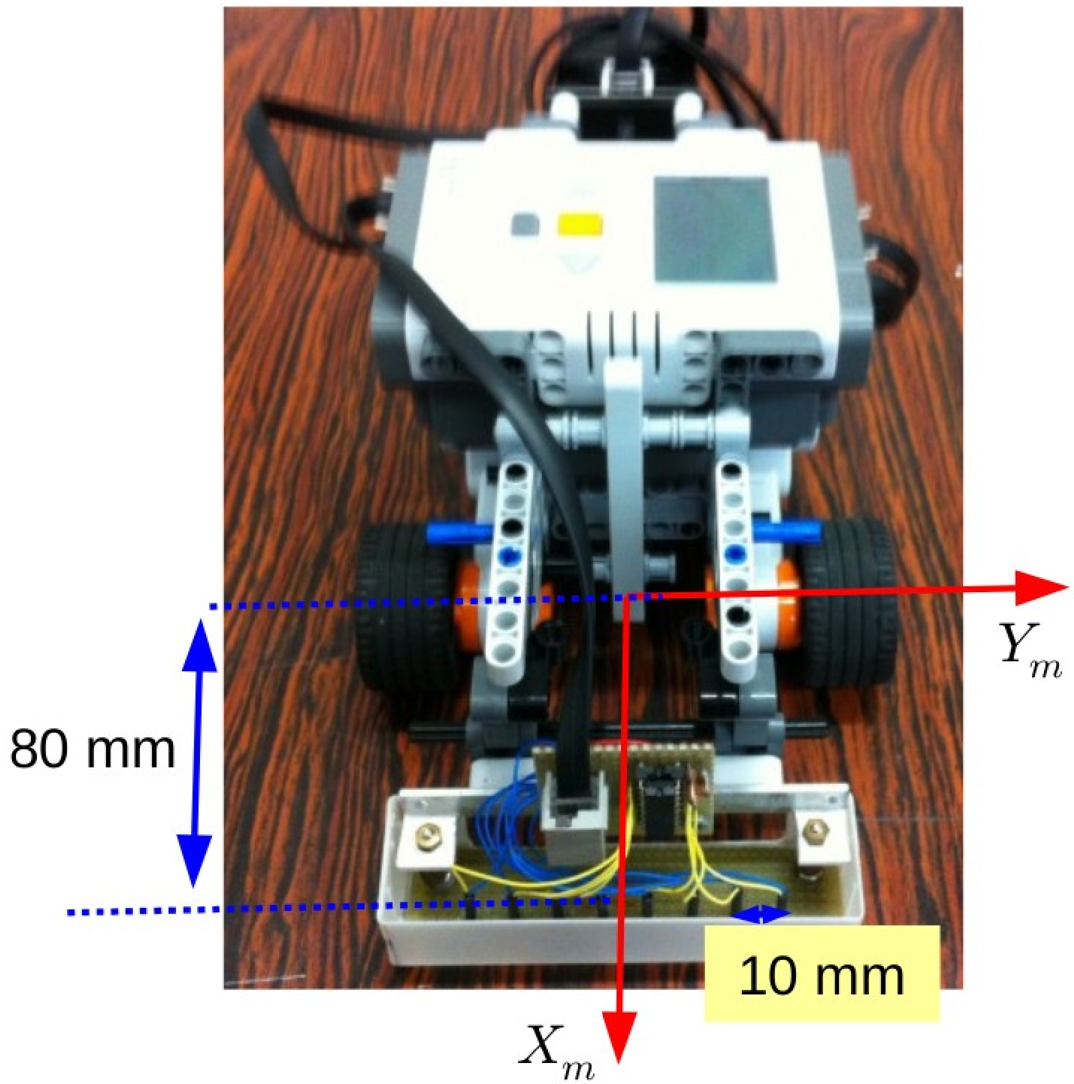
- The system design parameters are:
  - Magnet field intensity.
  - Inter-magnet distance
  - Reed switch sensitivity.
  - Reed switch number/spacing and sensor length.
- The sensor has been designed in such a way that:
  - When passing over a magnet, either one or two reed switches are activated.
  - When the robot moves, the sensor “cannot avoid crossing magnets”

System characteristics

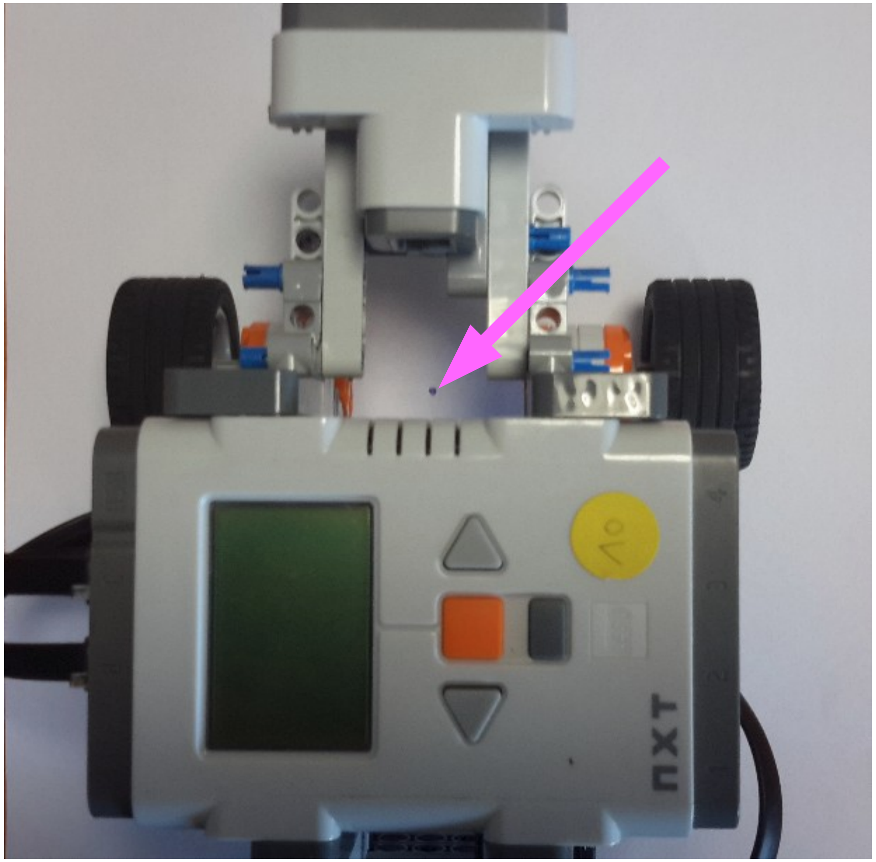
- 8 reed switches.
- 10 mm interval between reed switches
- 55 mm interval between magnets, arranged in a regular square pattern.



Robot and sensor



Initial positioning of the robot

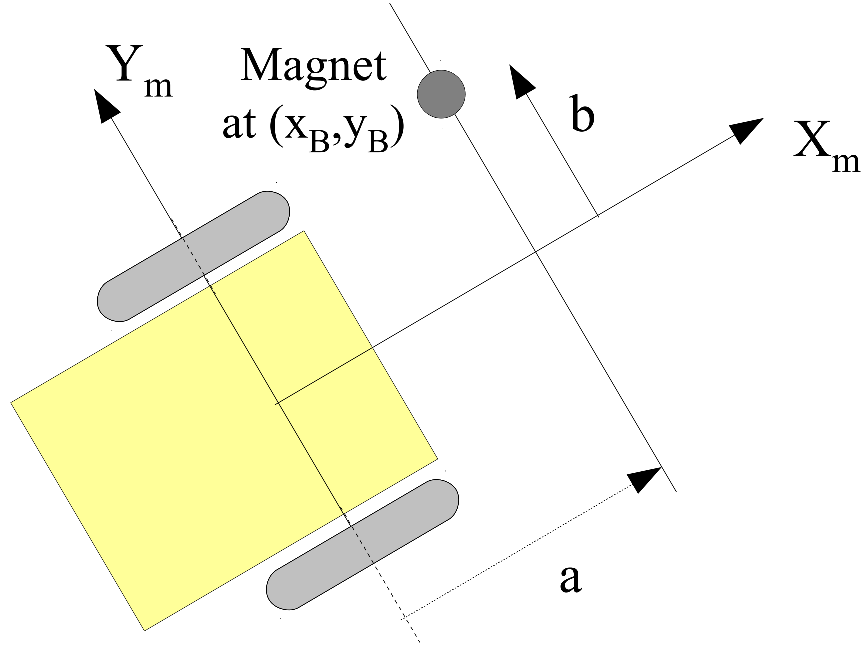


The center point of the fixed wheel axle
is put above a dot painted at (0,0).

Robot characteristics

- Encoders:
  - 360 dots per wheel revolution.
  - “Dumbed down” to 45 dots per revolution to make the problem a bit more challenging.
- Recording frequency:
  - 20 Hz.
  - We will work at 5 Hz.

The robot detecting a magnet



- Express in plain English what the sensor measures.
- Write the measurement equation

Variant of Kalman filter equations

- The program uses a slight variant of the Kalman filter equations, where the input u is assumed to be disturbed by noise:

$$X_{k+1} = f(X_k, U_k^*) + \alpha_k$$

$$U_k^* = U_k + \beta_k$$ The measured input is affected by an additive noise

$$P_{k+1} = A_k\, P_k\, A_k^T + B_k\, Q_\beta\, B_k^T + Q_\alpha$$

The error propagation equation is the only change.

$$A_k = \frac{\partial f}{\partial X} \qquad B_k = \frac{\partial f}{\partial U}$$

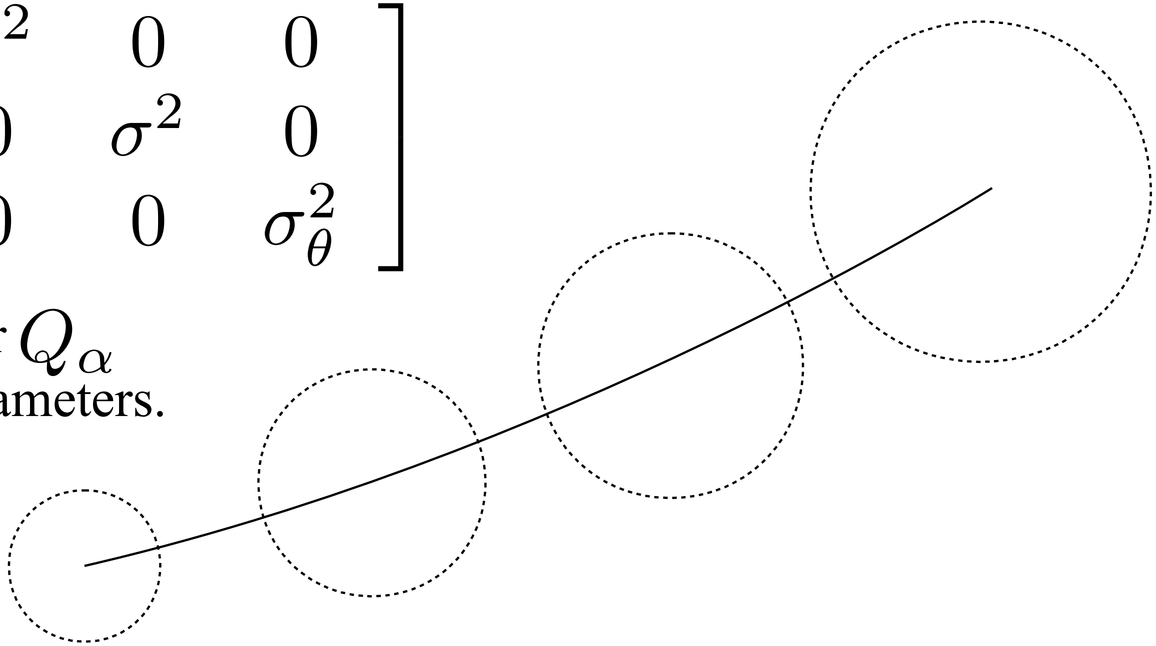
See paragraph 5.2 of the "book" form of the localization class material for the equations.

Evolution of uncertainty (standard form of the equations)

$$P_{k+1} = A_k \, P_k \, A_k^T + Q_\alpha$$

$$Q_\alpha = \begin{bmatrix} \sigma^2 & 0 & 0 \\ 0 & \sigma^2 & 0 \\ 0 & 0 & \sigma_\theta^2 \end{bmatrix}$$

A logical for for Q_α
Two tuning parameters.



Assuming the initial uncertainty is the same in x and y, the uncertainty ellipse
in the x-y plane starts as a circle and remains a circle during successive odometry phases (no update phase).

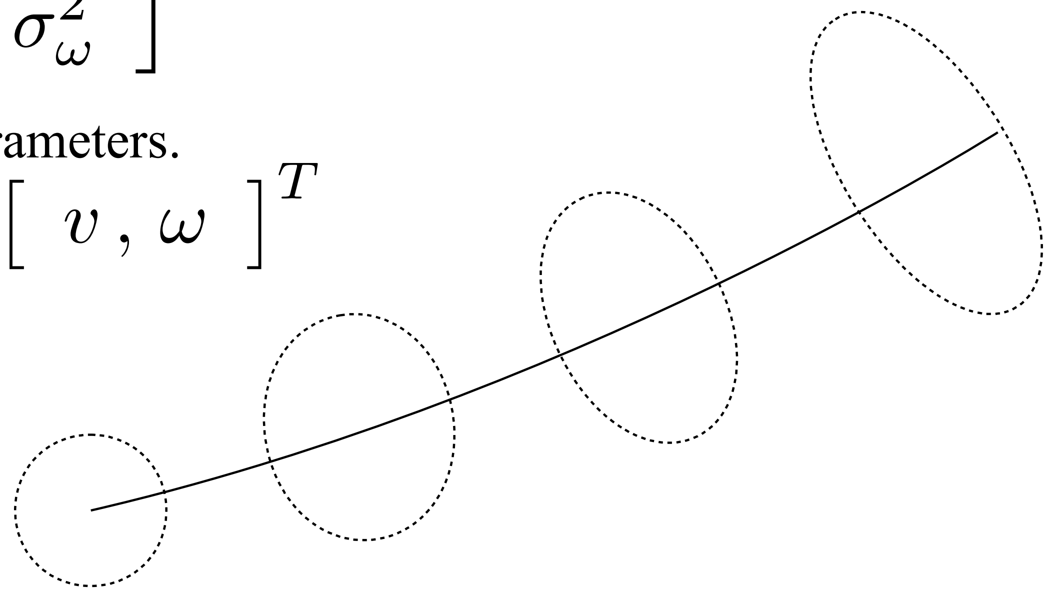
Evolution of uncertainty (form with noisy input)

$$P_{k+1} = A_k \, P_k \, A_k^T + B_k \, Q_\beta \, B_k^T$$ (Q_α has been set to zero)

$$Q_\beta = \begin{bmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_\omega^2 \end{bmatrix}$$

Still two tuning parameters.

Remember: $u = \begin{bmatrix} v \, , \, \omega \end{bmatrix}^T$



Now the uncertainty ellipse orients with the motion of the robot. The result is closer to the way errors actually evolve during odometry.

The input noise in the lab

$$\left\{ \begin{array}{l} v = (r_r \dot{q}_r + r_l \dot{q}_l)/2 \\ \omega = (r_r \dot{q}_r - r_l \dot{q}_l)/e \end{array} \right.$$

There is a linear relation between v, w and the rotation speed of the wheels.

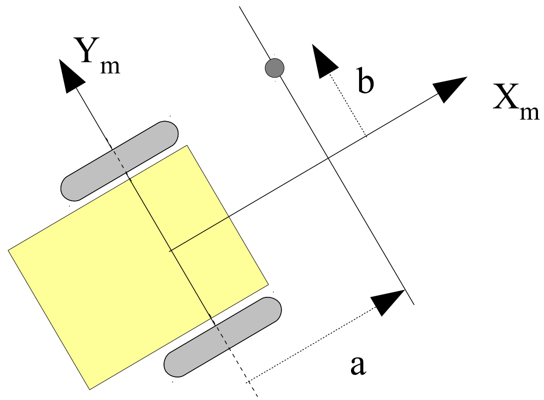
So Q_β can be written as a function of the covariance matrix of errors on $\dot{q}_r$ and $\dot{q}_l$.

$$Q_\beta = K\, Q_{\dot{q}}\, K^T \qquad \text{with} \qquad K = \begin{bmatrix} r_r/2 & r_l/2 \\ r_r/e & -r_l/e \end{bmatrix}$$

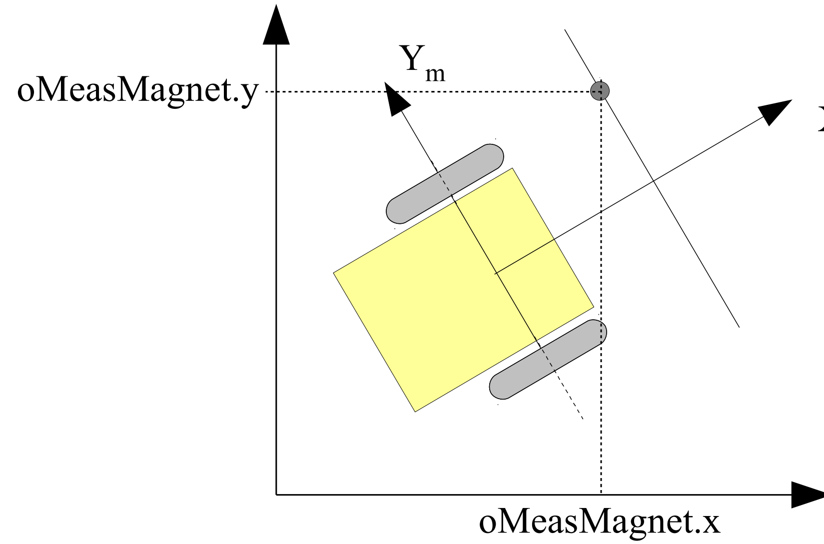
A reasonable form for $Q_{\dot{q}}$:

$$Q_{\dot{q}} = \begin{bmatrix} \sigma_{\dot{q}}^2 & 0 \\ 0 & \sigma_{\dot{q}}^2 \end{bmatrix}$$

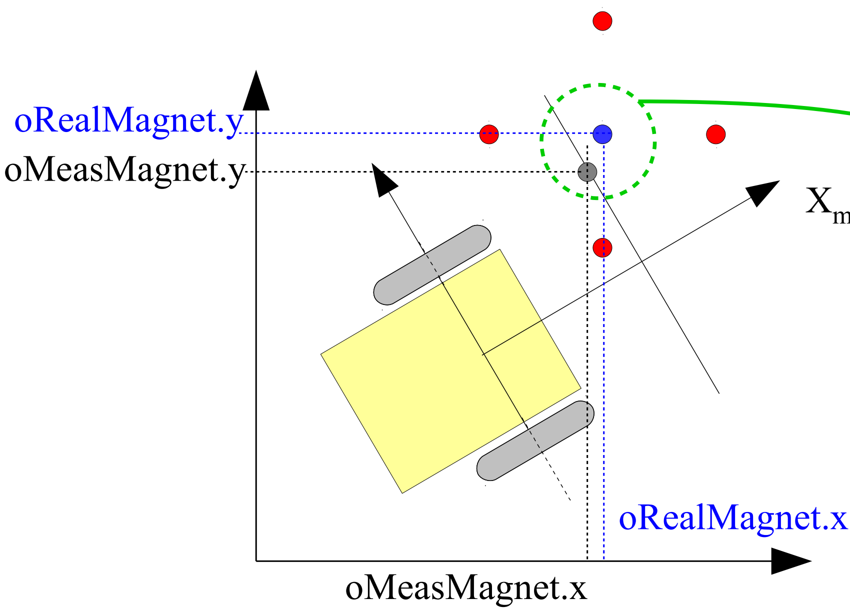
Now the number of tuning parameters for the input noise is 1.



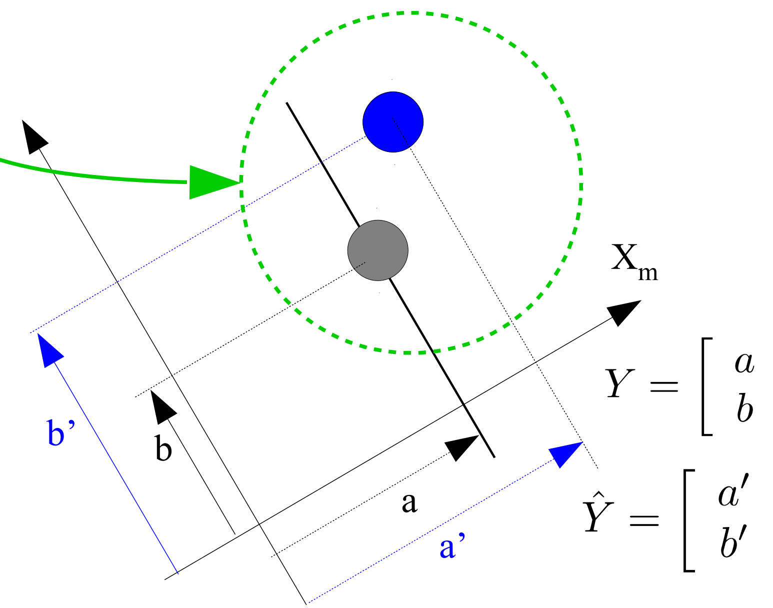
Step 1: measurement vector Y



Step 2: oMeasMagnet: coordinates of the measurement point in absolute frame



Step 3: No magnet coordinates exactly match, but one of them is closest: it's the candidate magnet. The algorithm assumes it's the correct one, and it will fail if not.



Step 4: The expected measurement is the coordinates of the real magnet in the robot frame