CSCE 574 ROBOTICS

Review

Ioannis Rekleitis

Fundamental Problems In Robotics

- What does the world looks like? (mapping)
  - sense from various positions
  - integrate measurements to produce map
  - assumes perfect knowledge of position
- Where am I in the world? (localization)
  - Sense
  - relate sensor readings to a world model
  - compute location relative to model
  - assumes a perfect world model
- Together, these are SLAM (Simultaneous Localization and Mapping)

Sensors

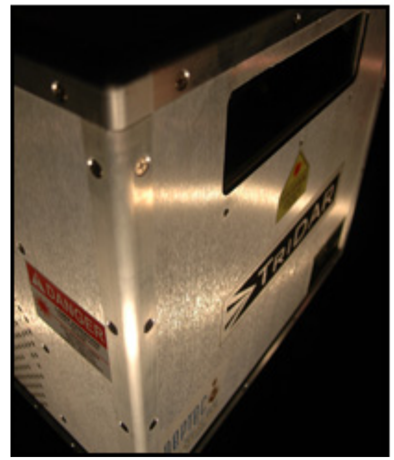
- **Proprioceptive Sensors**
  (monitor state of robot)
  - IMU (accels & gyros)
  - Wheel encoders
  - Doppler radar …

- **Exteroceptive Sensors**
  (monitor environment)
  - Cameras (single, stereo, omni, FLIR …)
  - Laser scanner
  - MW radar
  - Sonar
  - Tactile…

Types of sensor

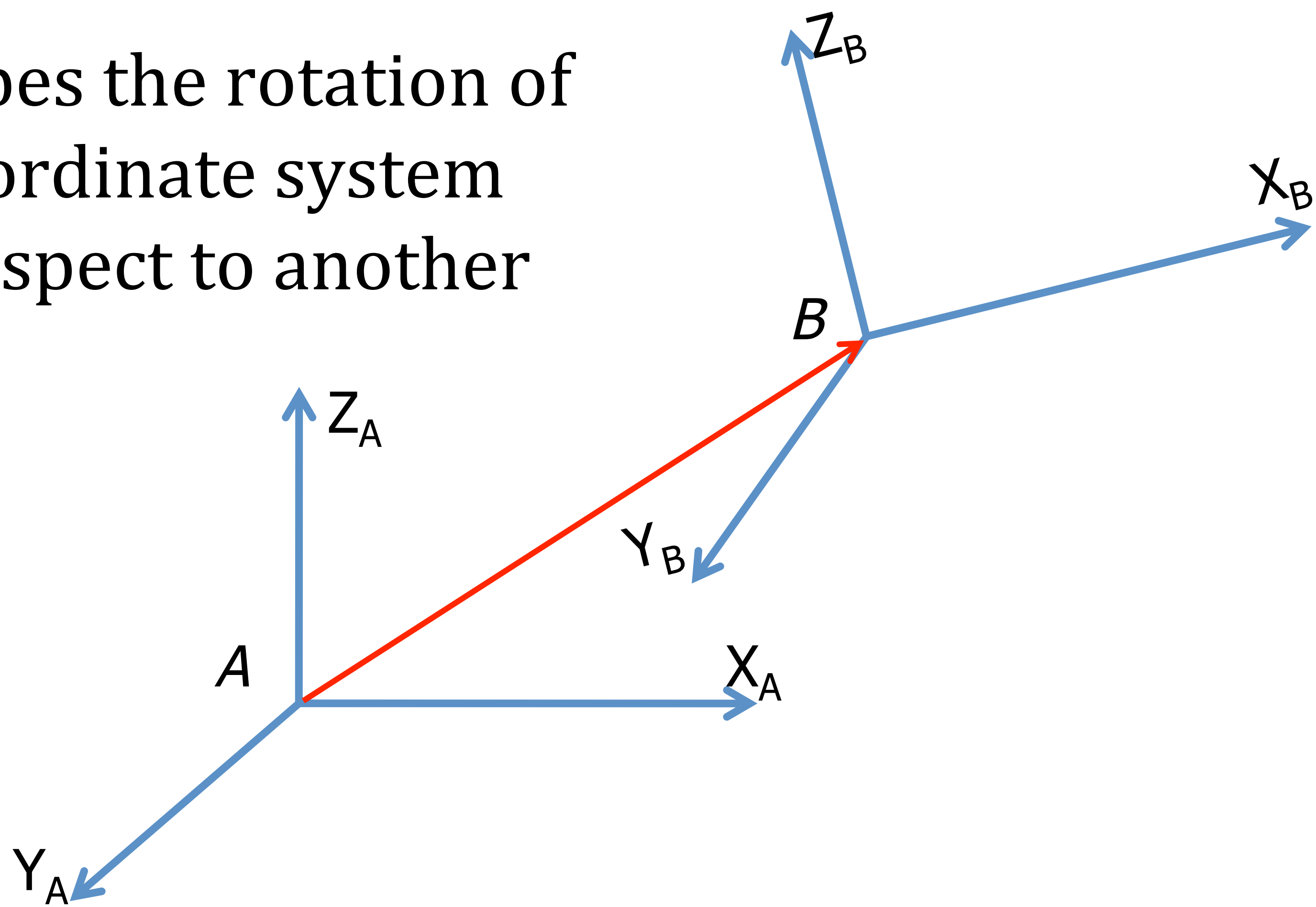
Specific examples

- tactile
- close-range proximity
- angular position
- infrared
- Sonar
- laser (various types)
- radar
- compasses, gyroscopes
- Force
- GPS
- vision



Orientation Representations

- Describes the rotation of one coordinate system with respect to another



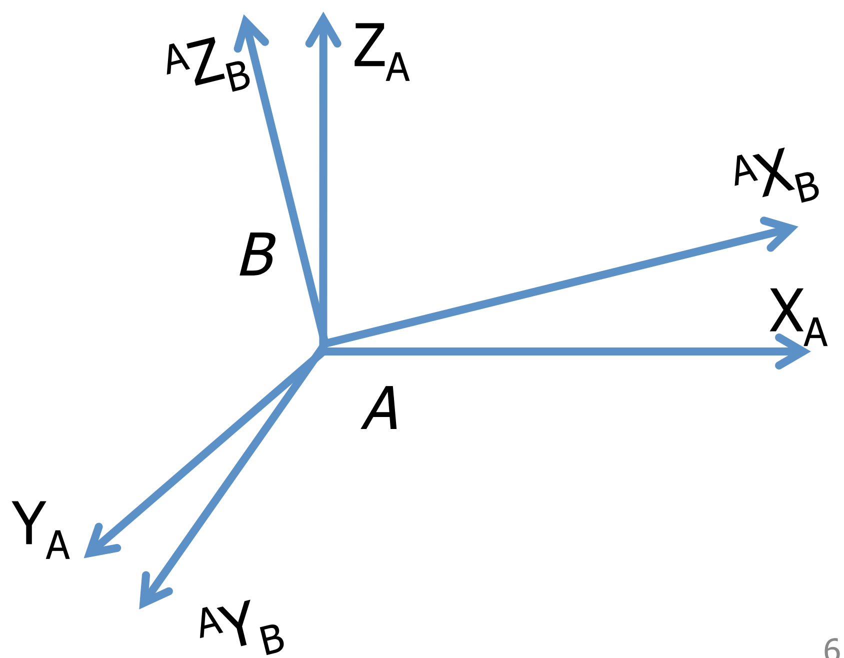
Attitude Representations

- Rotation Matrix: (9 variables)

$$ {}^{A}_{B}R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} $$

- Euler Angles [roll, pitch, yaw]
- (Gimbal Lock, 0-360 discontinuity, multiple representations)
- Angle-Axis [V,θ]
- Quaternions

$$ \bar{q} = q_4 + q_1 i + q_2 j + q_3 k $$



Actuators

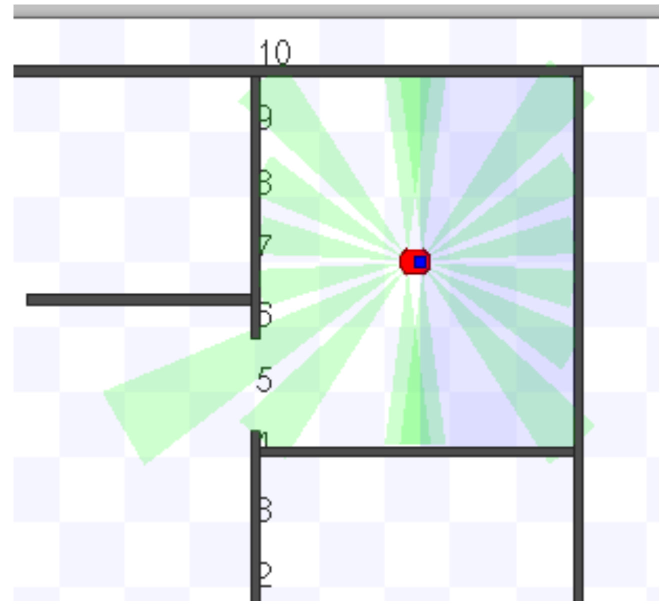
- Hydraulic Actuators
- Pneumatic Actuators
- Air Muscle
- Shape Memory Alloy Actuators
- Electric Actuators
- Stepper Motors

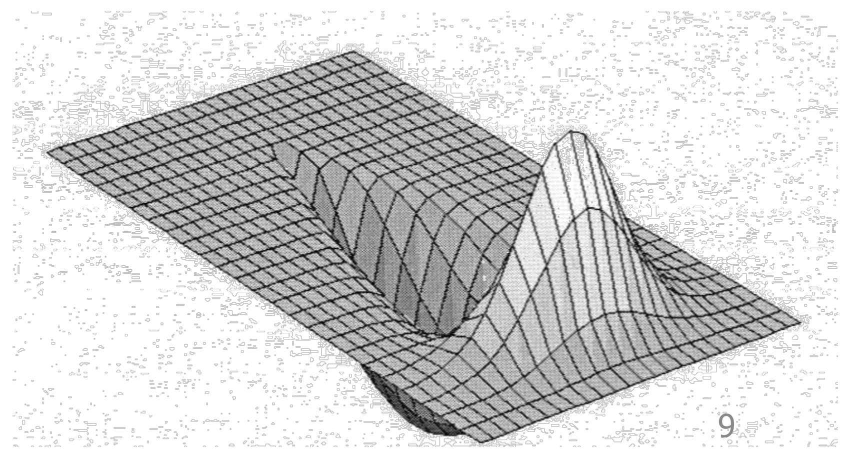
Locomotion

- Differential drive
- Synchronous drive
- Ackerman drive
- Legged Locomotion
  - Quadrupeds
  - Hexapod
  - Biped

Mapping

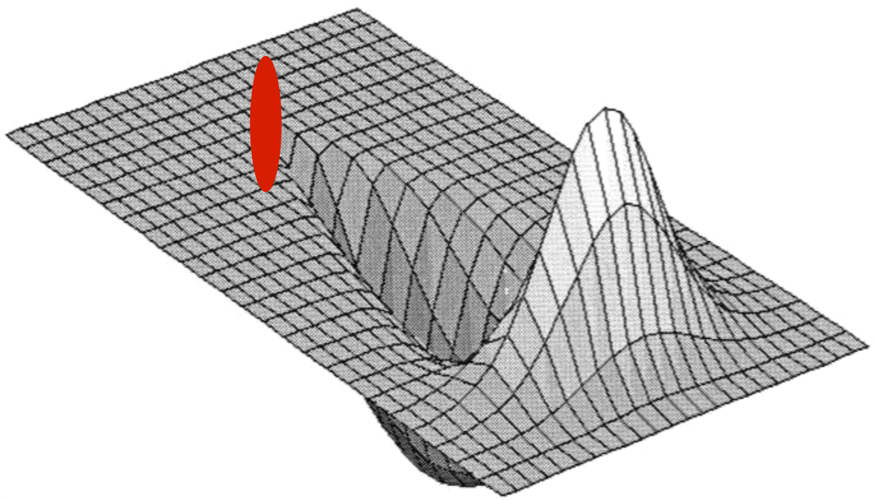
- Occupancy Grids
- Sonar model





SONAR modeling using Occupancy Grids

- The key to making accurate maps is combining lots of data.
- But combining these numbers means we have to know what they are !



what is in each cell of this sonar model / map ?

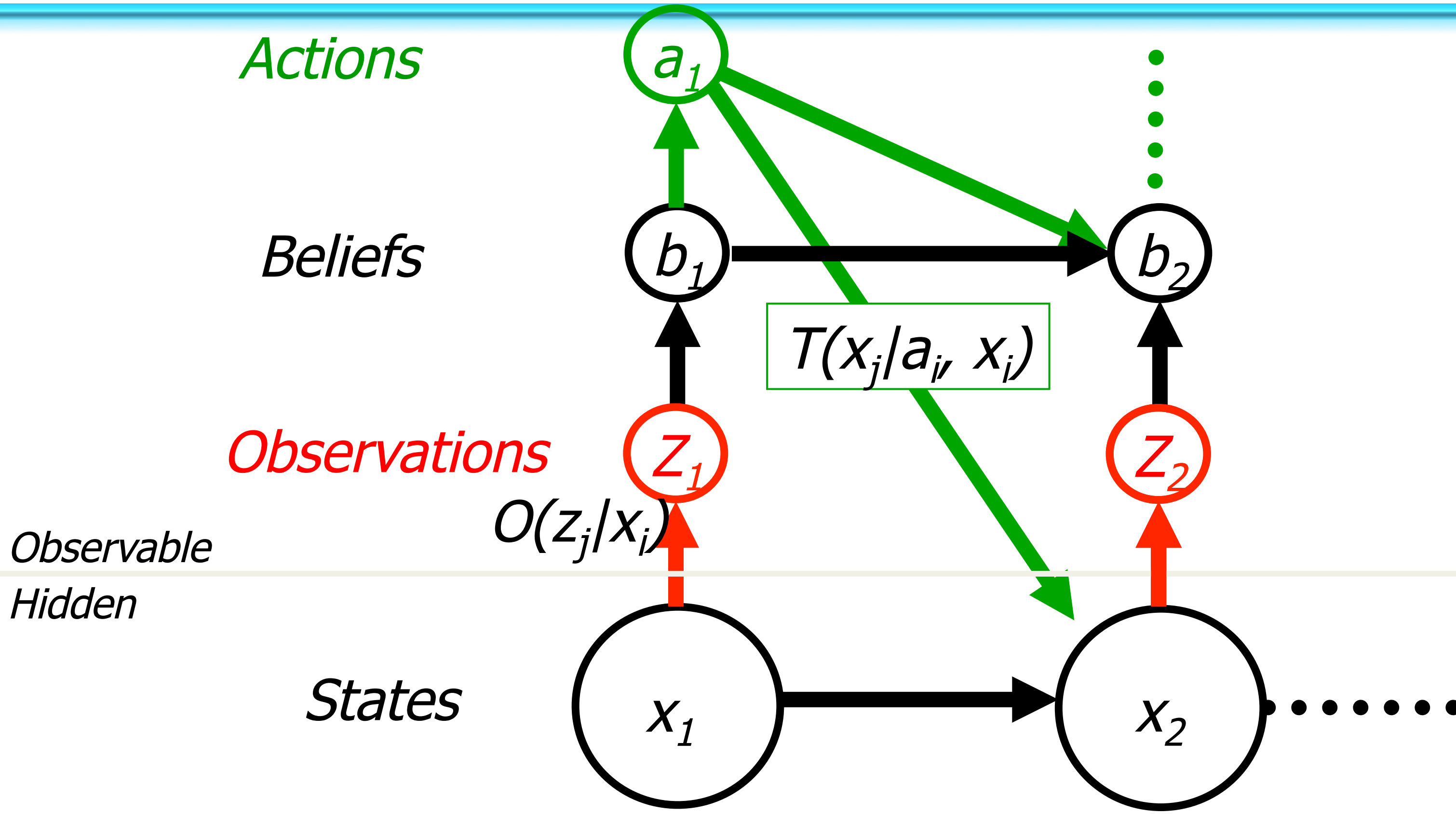
What should our map contain ?

- small cells
- each represents a bit of the robot's environment
- larger values => obstacle
- smaller values => free

Localization

- Tracking: Known initial position
- Global Localization: Unknown initial position
- Re-Localization: Incorrect known position
  - (kidnapped robot problem)

Graphical Models, Bayes' Rule and the Markov Assumption



$$\text{Bayes rule}: p(x \mid y) = \frac{p(y \mid x)\,p(x)}{p(y)}$$

$$\text{Markov}: p(x_t \mid x_{t-1}, a_t, a_0, z_0, a_1, z_1, \ldots, z_{t-1}) = p(x_t \mid x_{t-1}, a_t)$$

Derivation of the Bayesian Filter

First-order Markov assumption shortens middle term:

$$Bel(x_t) = \eta \, \boxed{p(o_t \mid x_t)} \int p(x_t \mid x_{t-1}, a_{t-1}) p(x_{t-1} \mid a_{t-1}, \ldots, o_0) \, dx_{t-1}$$

Finally, substituting the definition of *Bel(x*$_{t-1}$*):*

$$Bel(x_t) = \eta p(o_t \mid x_t) \int p(x_t \mid x_{t-1}, a_{t-1}) Bel(x_{t-1}) \, dx_{t-1}$$

The above is the probability distribution that must be estimated from the robot's data

Iterating the Bayesian Filter

- Propagate the motion model:

$$Bel\_(x_t) = \int P(x_t \mid a_{t-1}, x_{t-1}) Bel(x_{t-1}) dx_{t-1}$$

Compute the current state estimate before taking a sensor reading by integrating over all possible previous state estimates and applying the motion model

- Update the sensor model:

$$Bel(x_t) = \eta P(o_t \mid x_t) Bel\_(x_t)$$

Compute the current state estimate by taking a sensor reading and multiplying by the current estimate based on the most recent motion history

Different Approaches

Kalman filters (late-60s?)
- Gaussians
- approximately linear models
- position tracking

Extended Kalman Filter
Information Filter
Unscented Kalman Filter

Multi-hypothesis ('00)
- Mixture of Gaussians
- Multiple Kalman filters
- Global localization, recovery

Discrete approaches ('95)
- Topological representation ('95)
- uncertainty handling (POMDPs)
- occas. global localization, recovery
- Grid-based, metric representation ('96)
- global localization, recovery

Particle filters ('98)
- Condensation (Isard and Blake '98)
- Sample-based representation
- Global localization, recovery
- Rao-Blackwellized Particle Filter

Monte-Carlo State Estimation

(Particle Filtering)

- Employing a Bayesian Monte-Carlo simulation technique for pose estimation.
- A particle filter uses N samples as a discrete representation of the probability distribution function (*pdf*) of the variable of interest:

$$S = [\vec{\mathbf{x}}_i, w_i : i = 1 \cdots N]$$

where $\mathbf{x}_i$ is a copy of the variable of interest and w_i is a weight signifying the quality of that sample.

In our case, each particle can be regarded as an alternative hypothesis for the robot pose.

Particle Filter (cont.)

The particle filter operates in two stages:

- **Prediction:** After a motion (α) the set of particles S is modified according to the action model

$$S' = f(S, \alpha, \nu)$$

where (ν) is the added noise.

The resulting *pdf* is the prior estimate before collecting any additional sensory information.

Particle Filter (cont.)

- **Update:** When a sensor measurement (z) becomes available, the weights of the particles are updated based on the likelihood of (z) given the particle x_i

$$w_i' = P(z \mid \vec{\mathbf{x}}_i) w_i$$

The *updated particles* represent the posterior distribution of the moving robot.

Resampling

For finite particle populations, we must focus population mass where the *PDF* is substantive.

- Failure to do this correctly can lead to divergence.
- Resampling needlessly also has disadvantages.

One way is to estimate the need for resampling based on the variance of the particle weight distribution, in particular the coefficient of variance:

$$cv_t^2 = \frac{\mathrm{var}(w_t(i))}{E^2(w_t(i))} = \frac{1}{M}\sum_{i=1}^{M}(Mw_t(i)-1)^2$$

$$ESS_t = \frac{M}{1+cv_t^2}$$

The Kalman Filter

- Motion model is linear, noise is Gaussian…
- Sensor model is linear, noise is Gaussian…
- Each belief function is uniquely characterized by its mean μ and covariance matrix Σ
- Computing the posterior means computing a new mean μ and covariance Σ from old data using actions and sensor readings
- *What are the key limitations?*

  1) Unimodal distribution

  2) Linear assumptions

What we know…
What we don't know…

- We know what the control inputs of our process are
  - We know what we've told the system to do and have a model for what the expected output should be if everything works right
- We don't know what the noise in the system truly is
  - We can only estimate what the noise might be and try to put some sort of upper bound on it
- When estimating the state of a system, we try to find a set of values that comes as close to the truth as possible
  - There will always be some mismatch between our estimate of the system and the true state of the system itself. We just try to figure out how much mismatch there is and try to get the best estimate possible

Kalman Filter Components

(also known as: Way Too Many Variables…)

Linear discrete time dynamic system (motion model)

State — Control input — Process noise

$$x_{t+1} = F_t x_t + B_t u_t + G_t w_t$$

State transition function — Control input function — Noise input function with covariance Q

Measurement equation (sensor model)

Sensor reading — State — Sensor noise with covariance R

$$z_{t+1} = H_{t+1} x_{t+1} + n_{t+1}$$

Sensor function

Note:Write these down!!!

Computing the MMSE Estimate of the State and Covariance

What is the **minimum mean square error** estimate of the system state and covariance?

$$\hat{x}_{t+1|t} = F_t \hat{x}_{t|t} + B_t u_t$$ Estimate of the state variables

$$\hat{z}_{t+1|t} = H_{t+1} \hat{x}_{t+1|t}$$ Estimate of the sensor reading

$$P_{t+1|t} = F_t P_{t|t} F_t^T + G_t Q_t G_t^T$$ Covariance matrix for the state

$$S_{t+1|t} = H_{t+1} P_{t+1|t} H_{t+1}^T + R_{t+1}$$ Covariance matrix for the sensors

The Kalman Filter…

Propagation (motion model):

$$\hat{x}_{t+1/t} = F_t \hat{x}_{t/t} + B_t u_t$$

$$P_{t+1/t} = F_t P_{t/t} F_t^T + G_t Q_t G_t^T$$

Update (sensor model):

$$\hat{z}_{t+1} = H_{t+1} \hat{x}_{t+1/t}$$

$$r_{t+1} = z_{t+1} - \hat{z}_{t+1}$$

$$S_{t+1} = H_{t+1} P_{t+1/t} H_{t+1}{}^T + R_{t+1}$$

$$K_{t+1} = P_{t+1/t} H_{t+1}{}^T S_{t+1}{}^{-1}$$

$$\hat{x}_{t+1/t+1} = \hat{x}_{t+1/t} + K_{t+1} r_{t+1}$$

$$P_{t+1/t+1} = P_{t+1/t} - P_{t+1/t} H_{t+1}{}^T S_{t+1}{}^{-1} H_{t+1} P_{t+1/t}$$

…but what does that mean in English?!?

Propagation (motion model):

$$\hat{x}_{t+1/t} = F_t \hat{x}_{t/t} + B_t u_t$$
- State estimate is updated from system dynamics

$$P_{t+1/t} = F_t P_{t/t} F_t^T + G_t Q_t G_t^T$$
- Uncertainty estimate *GROWS*

Update (sensor model):

$$\hat{z}_{t+1} = H_{t+1} \hat{x}_{t+1/t}$$
- Compute expected value of sensor reading

$$r_{t+1} = z_{t+1} - \hat{z}_{t+1}$$
- Compute the difference between expected and "true"

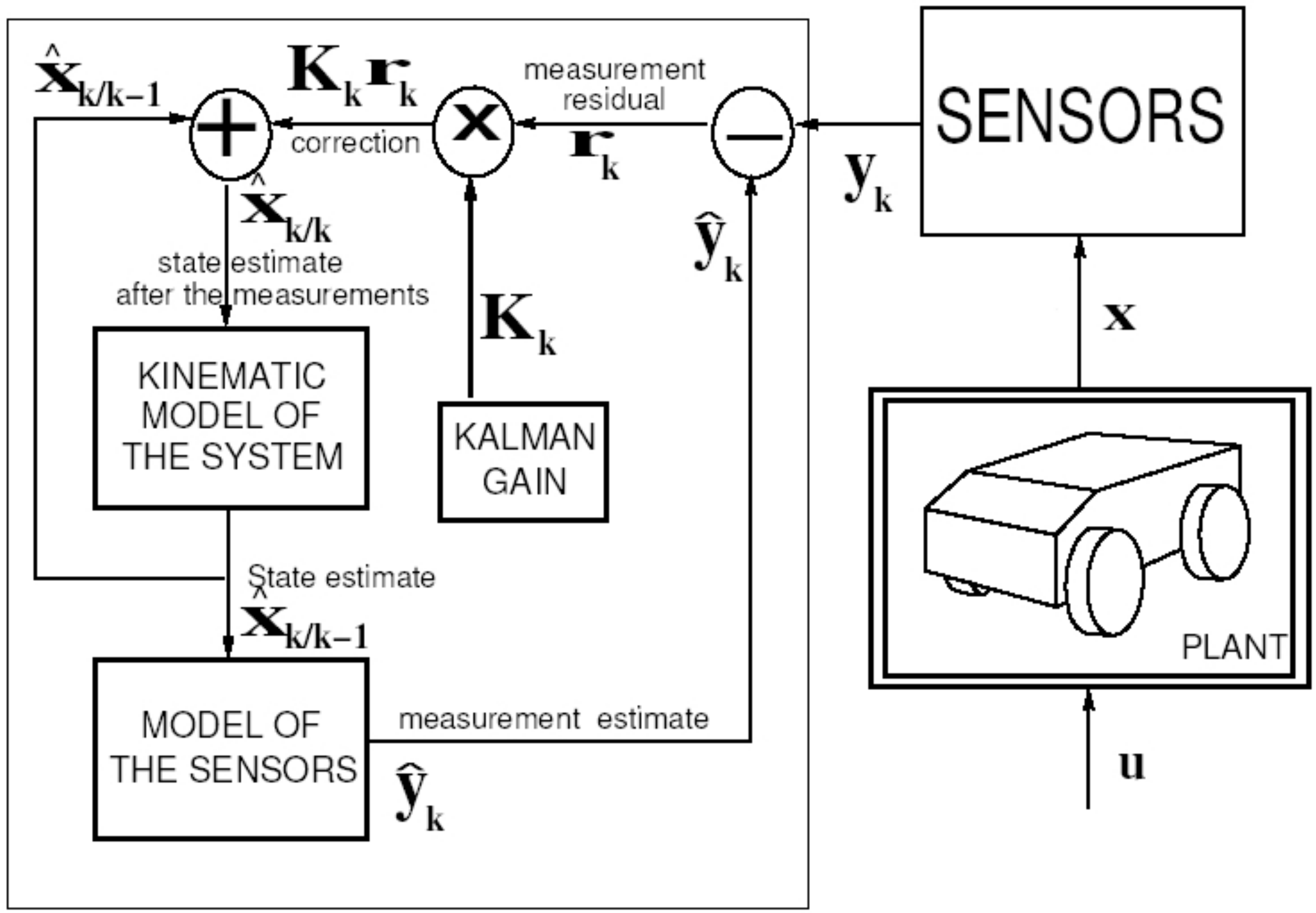
$$S_{t+1} = H_{t+1} P_{t+1/t} H_{t+1}{}^T + R_{t+1}$$
- Compute covariance of sensor reading

$$K_{t+1} = P_{t+1/t} H_{t+1}{}^T S_{t+1}{}^{-1}$$
- Compute the Kalman Gain (how much to correct est.)

$$\hat{x}_{t+1/t+1} = \hat{x}_{t+1/t} + K_{t+1} r_{t+1}$$
- Multiply residual times gain to correct state estimate

$$P_{t+1/t+1} = P_{t+1/t} - P_{t+1/t} H_{t+1}{}^T S_{t+1}{}^{-1} H_{t+1} P_{t+1/t}$$
- Uncertainty estimate SHRINKS

Kalman Filter Block Diagram



Calculation of $\tilde{\phi}_{t+1}$

$$
\begin{aligned}
\tilde{\phi}_{t+1} &= \phi_{t+1} - \hat{\phi}_{t+1} \\
&= \phi_t + \omega_t \Delta t - \hat{\phi}_t - (\omega_t + w_\omega)\Delta t \\
&= \tilde{\phi}_t - w_\omega \Delta t
\end{aligned}
$$

Calculation of $\tilde{x}_{t+1} = x_{t+1} - \hat{x}_{t+1}$ $\tilde{y}_{t+1} = y_{t+1} - \hat{y}_{t+1}$

$$\tilde{x}_{t+1} = x_{t+1} - \hat{x}_{t+1}$$

$$\begin{aligned}
\tilde{x}_{t+1} &= x_{t+1} - \hat{x}_{t+1} \\
&= x_t + v_t \Delta t \cos(\phi_t) - \hat{x}_t - (v_t - w_v)\Delta t \cos(\hat{\phi}_t) \\
&= x_t - \hat{x}_t + v_t \Delta t \cos(\phi_t) - v_t \Delta t \cos(\hat{\phi}_t) + w_v \Delta t \cos(\hat{\phi}_t) \\
&= \tilde{x}_t + v_t \Delta t \cos(\tilde{\phi}_t + \hat{\phi}_t) - v_t \Delta t \cos(\hat{\phi}_t) + w_v \Delta t \cos(\hat{\phi}_t) \\
&= \tilde{x}_t + v_t \Delta t [\cos(\tilde{\phi}_t)\cos(\hat{\phi}_t) + \sin(\tilde{\phi}_t)\sin(\hat{\phi}_t)] - v_t \Delta t \cos(\hat{\phi}_t) + w_v \Delta t \cos(\hat{\phi}_t) \\
&\cong \tilde{x}_t + v_t \Delta t \cos(\hat{\phi}_t) - v_t \Delta t \tilde{\phi}_t \sin(\hat{\phi}_t) - v_t \Delta t \cos(\hat{\phi}_t) + w_v \Delta t \cos(\hat{\phi}_t) \\
&= \tilde{x}_t - v_t \Delta t \tilde{\phi}_t \sin(\hat{\phi}_t) + w_v \Delta t \cos(\hat{\phi}_t)
\end{aligned}$$

$$\begin{aligned}
\tilde{y}_{t+1} &= y_{t+1} - \hat{y}_{t+1} \\
&= \ldots
\end{aligned}$$

Covariance Estimation

$$P_{t+1/t} = E[\widetilde{X}_{t+1}\widetilde{X}_{t+1}^T]$$

$$= E[(F_t\widetilde{X}_t + G_t w_t)(F_t\widetilde{X}_t + G_t w_t)^T]$$

$$= F_t E[\widetilde{X}_t\widetilde{X}_t^T]F_t^T + G_t E[w_t w_t^T]G_t^T$$

$$= F_t P_{t/t} F_t^T + G_t Q_t G_t^T$$

where

$$Q_t = E[w_t w_t^T] = \begin{bmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_\omega^2 \end{bmatrix}$$

Some observations

- The larger the error, the smaller the effect on the final state estimate
  - If *process* uncertainty is larger, *sensor* updates will dominate state estimate
  - If *sensor* uncertainty is larger, *process* propagation will dominate state estimate
- Improper estimates of the state and/or sensor covariance may result in a rapidly diverging estimator
  - As a rule of thumb, the residuals must always be bounded within a $\pm 3\sigma$ region of uncertainty
  - This measures the "health" of the filter
- *Many* propagation cycles can happen between updates