Distributed Intelligent Systems – W5

More on Localization Methods and an Introduction to Collective Movements

Outline

- Robot localization with uncertainties
  - fusion of proprioceptive and exteroceptive sensory 1D data for with Kalman filters
  - non-deterministic uncertainties in wheel-based encoders
  - multi-dimensional Kalman filters and localization in 2D

- Collective movements
  - Form of collective movements in animal societies
  - Flocking in virtual agents: Reynolds' Boids

The Kalman Filter Algorithm in 1D

Two Key Sources of Information

Stochastic models, estimation, and control

VOLUME 1

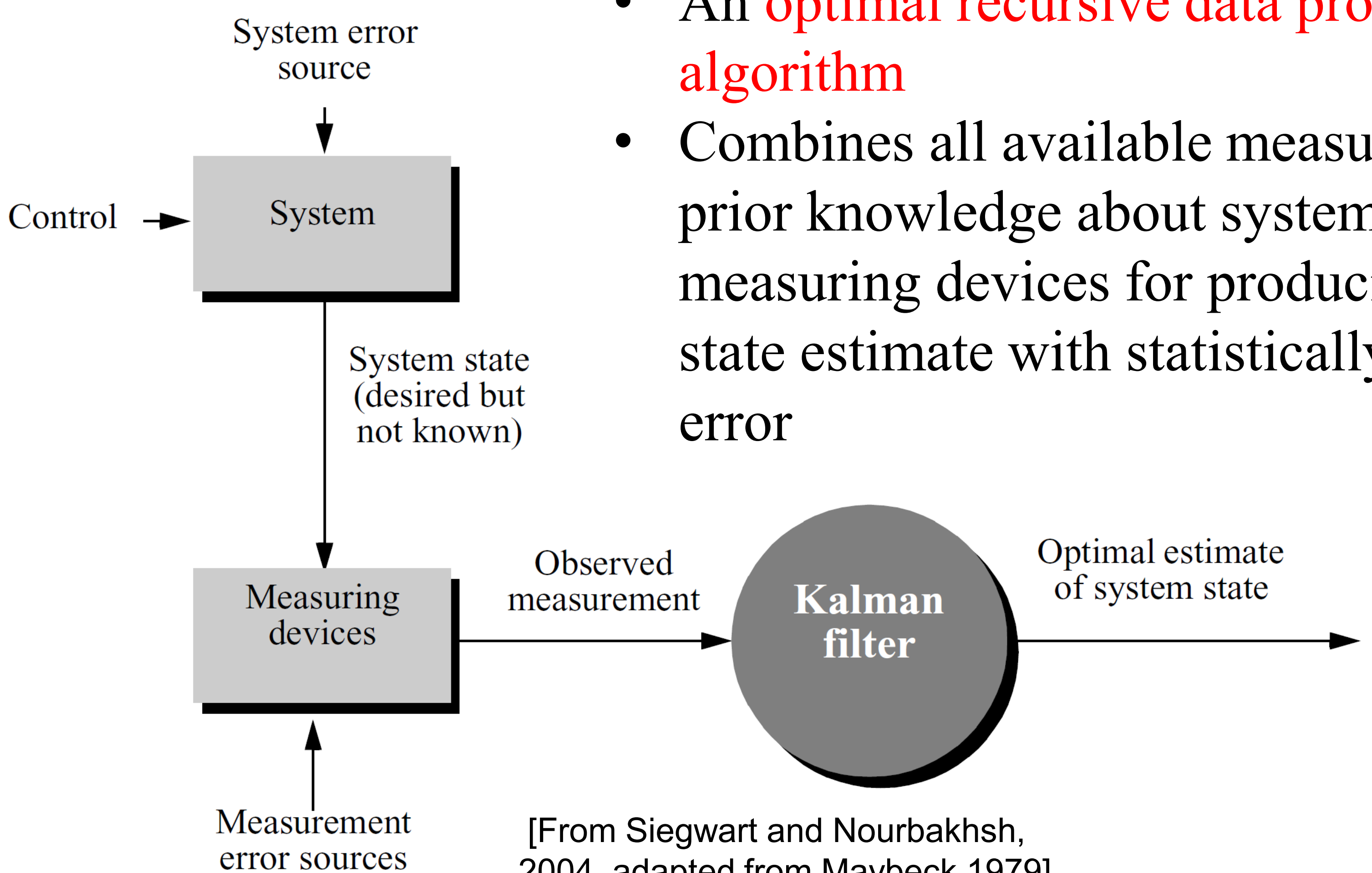
PETER S. MAYBECK

DEPARTMENT OF ELECTRICAL ENGINEERING
AIR FORCE INSTITUTE OF TECHNOLOGY
WRIGHT-PATTERSON AIR FORCE BASE
OHIO

ACADEMIC PRESS New York San Francisco London 1979
A Subsidiary of Harcourt Brace Jovanovich, Publishers

Kalman Filter - Overview

- An optimal recursive data processing algorithm
- Combines all available measurement data, prior knowledge about system and measuring devices for producing a system state estimate with statistically minimized error



[From Siegwart and Nourbakhsh, 2004, adapted from Maybeck 1979]

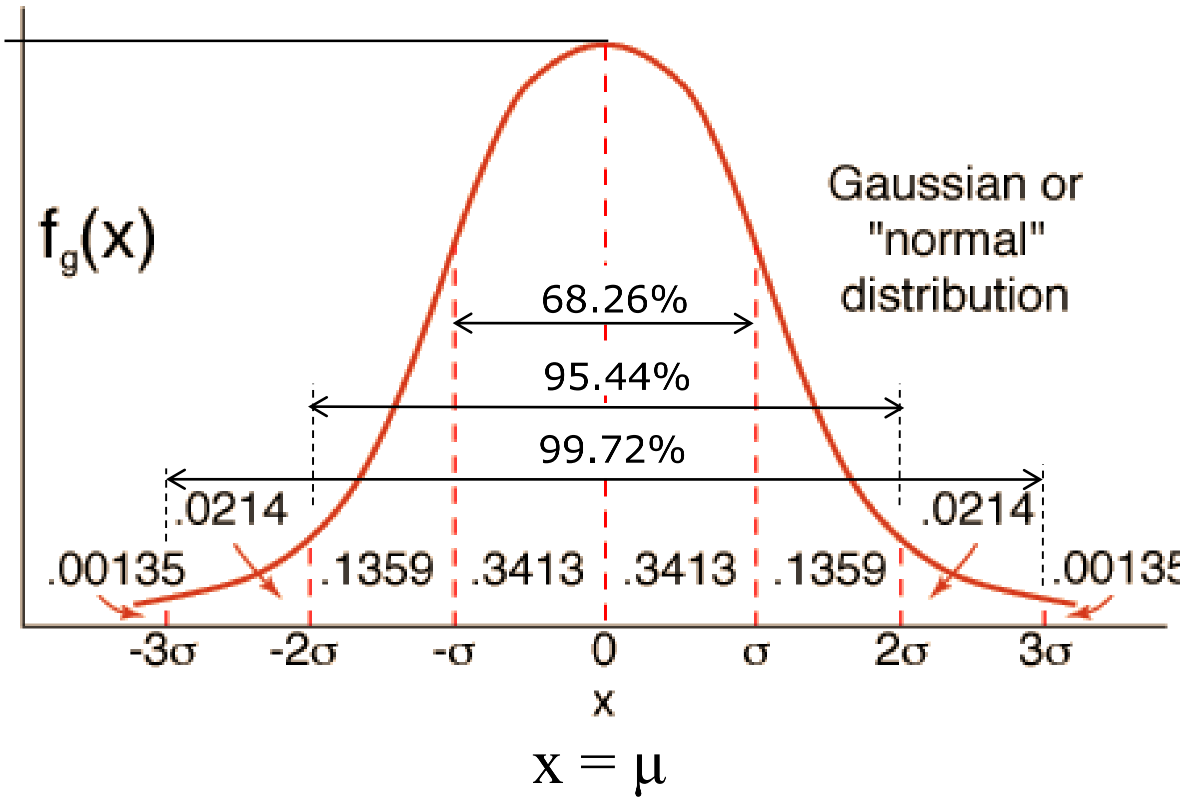
Kalman Filter - Assumptions

- Linear system model
- White Gaussian system and measurement noise

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

Notes on noise:

- White: uncorrelated in time
- Gaussian: probability density of amplitude follows bell-shaped curve



A Simple 1D Positioning Example

t = time

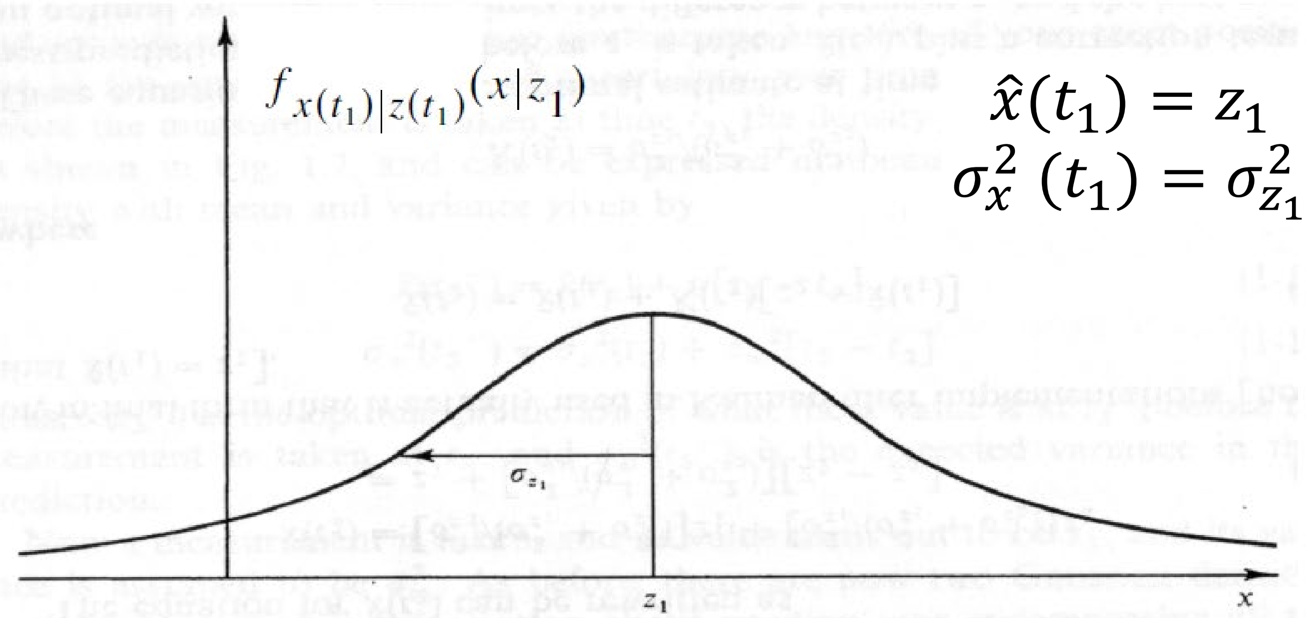
x = 1D position

$\hat{x}$ = 1D position estimate

z = measurement or observation

t_i = time instant i

z_i = measurement taken at t_i

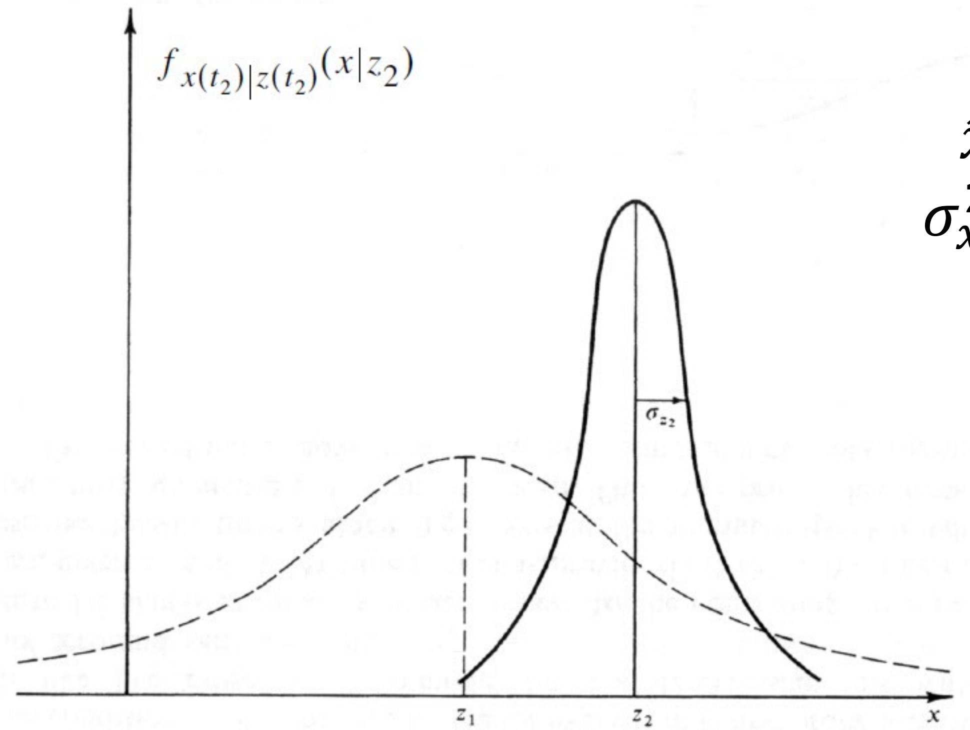


$$\hat{x}(t_1) = z_1$$
$$\sigma_x^2\,(t_1) = \sigma_{z_1}^2$$

[From Maybeck, 1979]

Estimation Based on Static Measurements

- Second measurement taken at $t_2 \approx t_1$
- The smaller σ, the higher the certitude about the measurement

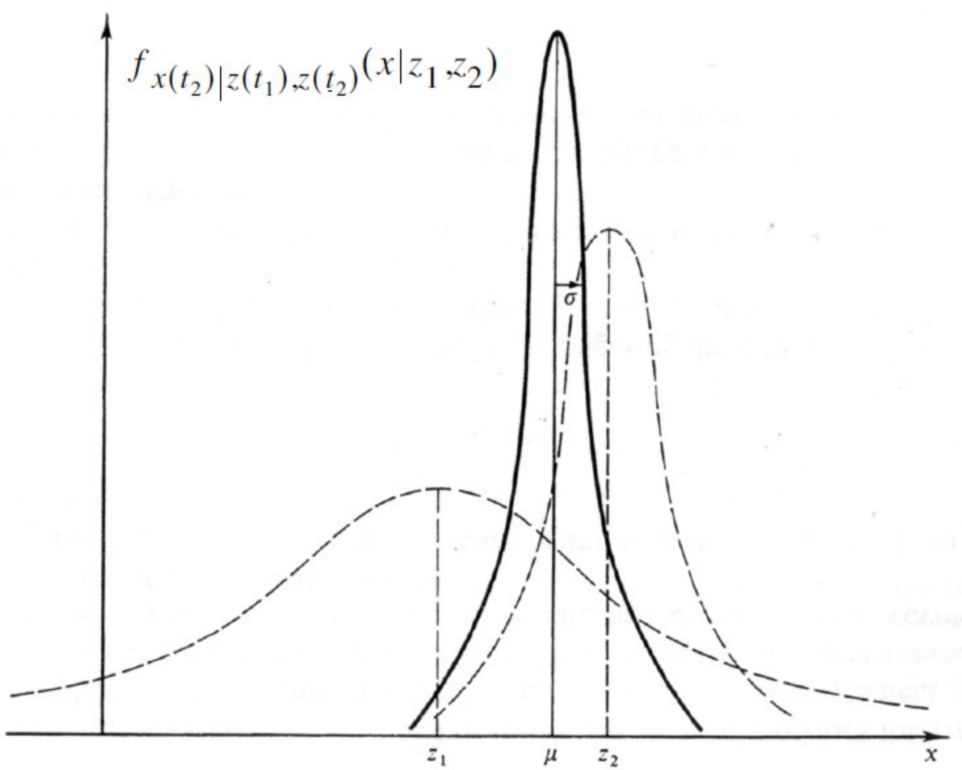


$$\hat{x}(t_2) = z_2$$
$$\sigma_x^2\,(t_2) = \sigma_{z_2}^2$$

[From Maybeck, 1979]

Improving the Estimate Through Fusion

- Intuition: the smaller σ, and thus σ^2, the higher should be the weight in the fused estimate
- Estimate can be obtained as a weighted average μ of the individual measurement contributions



[From Maybeck, 1979]

$$\mu = \frac{\frac{1}{\sigma_{z_1}^2} z_1 + \frac{1}{\sigma_{z_2}^2} z_2}{\frac{1}{\sigma_{z_1}^2} + \frac{1}{\sigma_{z_2}^2}}$$

$$\frac{1}{\sigma^2} = \frac{1}{\sigma_{z_1}^2} + \frac{1}{\sigma_{z_2}^2}$$

$$\hat{x}(t_2) = \mu$$

$$\sigma_x(t_2) = \sigma$$

$$\sigma < \sigma_{z_1} \text{ and } \sigma < \sigma_{z_2}$$

Mean of the new Estimate

$$\hat{x}(t_2) = \frac{\frac{1}{\sigma_{z_1}^2} z_1 + \frac{1}{\sigma_{z_2}^2} z_2}{\frac{1}{\sigma_{z_1}^2} + \frac{1}{\sigma_{z_2}^2}} = \frac{\sigma_{z_2}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2} z_1 + \frac{\sigma_{z_1}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2} z_2$$

$$= \frac{\sigma_{z_2}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2} z_1 \boxed{+ \frac{\sigma_{z_1}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2} z_1 - \frac{\sigma_{z_1}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2} z_1} + \frac{\sigma_{z_1}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2} z_2$$

$$= \frac{\sigma_{z_2}^2 + \sigma_{z_1}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2} z_1 + \frac{\sigma_{z_1}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2} (z_2 - z_1)$$

$$= z_1 + \frac{\sigma_{z_1}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2} (z_2 - z_1)$$

Kalman filter formulation

$$\hat{x}(t_2) = \hat{x}(t_1) + K(t_2)[z_2 - \hat{x}(t_1)]$$

with $K(t_2) = \frac{\sigma_{z_1}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2}$

$\hat{x}(t_1) = z_1$

Variance of the new Estimate

$$\frac{1}{\sigma_x^2(t_2)} = \frac{1}{\sigma_{z_1}^2} + \frac{1}{\sigma_{z_2}^2}$$

$$\sigma_x^2(t_2) = \frac{\sigma_{z_1}^2\sigma_{z_2}^2}{\sigma_{z_1}^2+\sigma_{z_2}^2} = \frac{\sigma_{z_2}^2}{\sigma_{z_1}^2+\sigma_{z_2}^2}\,\sigma_{z_1}^2 = \frac{\sigma_{z_1}^2+\sigma_{z_2}^2-\sigma_{z_1}^2}{\sigma_{z_1}^2+\sigma_{z_2}^2}\,\sigma_{z_1}^2 = \left(1-\frac{\sigma_{z_1}^2}{\sigma_{z_1}^2+\sigma_{z_2}^2}\right)\sigma_{z_1}^2$$

$$= \sigma_{z_1}^2 - \frac{\sigma_{z_1}^2}{\sigma_{z_1}^2+\sigma_{z_2}^2}\,\sigma_{z_1}^2$$

Kalman filter formulation

$$\sigma_x^2(t_2) = \sigma_x^2\ (t_1)\ - K(t_2)\sigma_x^2(t_1)$$

with $K(t_2) = \dfrac{\sigma_{z_1}^2}{\sigma_{z_1}^2+\sigma_{z_2}^2}$

$$\sigma_x^2(t_1) = \sigma_{z_1}^2$$

Kalman Filter for Sensor Fusion

$$\hat{x}(t_2) = \hat{x}(t_1) + K(t_2)[z_2 - \hat{x}(t_1)]$$

Mean

Prediction

Correction or Update

$$\sigma_x^2(t_2) = \sigma_x^2(t_1) - K(t_2)\,\sigma_x^2(t_1)$$

Variance

$$K(t_2) = \frac{\sigma_{z_1}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2}$$

Kalman Gain

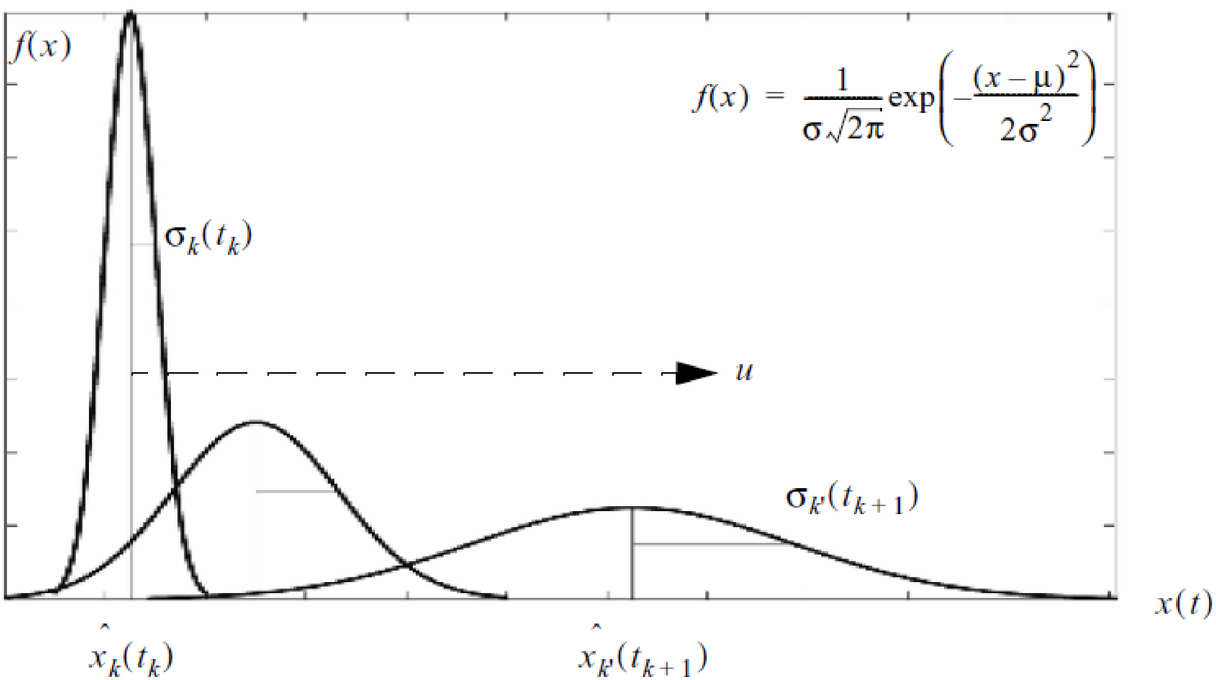
Function of the sensing precision

Estimation Considering Motion Dynamics

Consider a simple but noisy motion model: $\dot{x} = u + w$

u = constant speed (controllable input)
w = Gaussian motion noise

$w = \sigma_w^2$



[From Siegwart and Nourbakhsh, 2004, adapted from Maybeck 1979]

σ_k^2: variance at timestep k (known)

$\sigma_{k'}^2$: variance at timestep k+1

Estimation Based on Motion Model

$$\hat{x}_{k'} = \hat{x}_k + u(t_{k+1} - t_k)$$

- New mean position at timestep t_{k+1}
- Can be estimated with deterministic displacement from motion model

$$\sigma_{k'}^2 = \sigma_k^2 + \sigma_w^2(t_{k+1} - t_k)$$

- New variance at timestep t_{k+1}
- Variance of noisy motion (constant over time) gets added (cumulated) to previous one

Fusing Motion Model Prediction with New Measurement - Mean

$$\hat{x}_{k+1} = \hat{x}_{k'} + K_{k+1}(z_{k+1} - \hat{x}_{k'}) \qquad \text{with } K_{k+1} = \frac{\sigma_{k'}^2}{\sigma_{k'}^2 + \sigma_z^2}$$

Prediction based on motion model

Correction or update based on observation

z_{k+1} : measurement at timestep k+1

K_{k+1} : Kalman gain at timestep k+1

$\hat{x}_{k+1}$: new estimate at timestep k+ 1 incorporating observation and prediction of motion model

$\hat{x}_{k'}$: estimate just before timestep k+1 based on prediction motion model

Fusing Motion Model Prediction with New Measurement - Variance

$$\sigma_{k+1}^2 = \sigma_{k'}^2 - K_{k+1}\,\sigma_{k'}^2 \qquad \text{with } K_{k+1} = \frac{\sigma_{k'}^2}{\sigma_{k'}^2 + \sigma_z^2}$$

Prediction based on motion model

Correction or update based on observation

K_{k+1} : Kalman gain at timestep k+1

σ_{k+1}^2 : variance at timestep k+ 1 incorporating correction from observation and prediction of motion model

$\sigma_{k'}^2$: variance just before timestep k+1 based on prediction motion model

σ_z^2 : variance of the sensor meausurement (constant over time)

Kalman Filter for Sensor and Motion Model Fusion

$$\hat{x}_{k+1} = \hat{x}_{k'} + K_{k+1}(z_{k+1} - \hat{x}_{k'})$$

Mean

Prediction

Correction or Update

$$\sigma^2_{k+1} = \sigma^2_{k'} - K_{k+1}\,\sigma^2_{k'}$$

Variance

$$K_{k+1} = \frac{\sigma^2_{k'}}{\sigma^2_{k'} + \sigma^2_z}$$

Function of the motion model and sensing precision

Kalman Gain

Kalman Filter - Some Extreme Cases

$$\hat{x}_{k+1} = \hat{x}_{k'} + K_{k+1}(z_{k+1} - \hat{x}_{k'})$$

$$\sigma_{k+1}^2 = \sigma_{k'}^2 - K_{k+1}\,\sigma_{k'}^2$$

$$K_{k+1} = \frac{\sigma_{k'}^2}{\sigma_{k'}^2 + \sigma_z^2}$$

$\sigma_z^2 \rightarrow \infty$: new measurement extremely noisy, does not add information, then $K_{k+1} \rightarrow 0$ new estimate based exclusively on motion model (both mean and variance)

$\sigma_w^2 \rightarrow \infty$, then $\sigma_{k'}^2 \rightarrow \infty$ (see s. 14): motion model does not add information, then $K_{k+1} \rightarrow 1$ new estimate based exclusively on new observation

$\sigma_{k'}^2 \rightarrow 0$, motion model is deterministic and perfectly reproducing the reality, then $K_{k+1} \rightarrow 0$, new measurement can be disregarded since model is giving a perfect estimate

Wheel-Based Odometry in Practice

From Model to Practice

From Week 4, s. 28:

$$\dot{\xi}_I = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{r\dot{\varphi}_1}{2} + \frac{r\dot{\varphi}_2}{2} \\ 0 \\ \frac{r\dot{\varphi}_1}{2l} + \frac{-r\dot{\varphi}_2}{2l} \end{bmatrix}$$

$v_r = r\dot{\varphi}_1$ (right wheel speed)

$v_l = r\dot{\varphi}_2$ (left wheel speed)

$b = 2l$ (inter-wheel distance)

With x, y, and θ in the inertial frame:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{v_r + v_l}{2} \\ 0 \\ \frac{v_r - v_l}{b} \end{bmatrix}$$

From Model to Practice

$$\begin{cases} \dot{x} = \frac{v_r+v_l}{2}\cos\theta \\ \dot{y} = \frac{v_r+v_l}{2}\sin\theta \\ \dot{\theta} = \frac{v_r - v_l}{b} \end{cases}$$

- Assume small time interval Δt
- Assume v_r and v_l constant in time interval Δt
- Transform differential in difference equations and approximate over Δt
- $\theta = \theta$(t) -> rotational matrix in the middle of interval Δt -> $\tilde{\theta} = \theta + \Delta\theta/2$

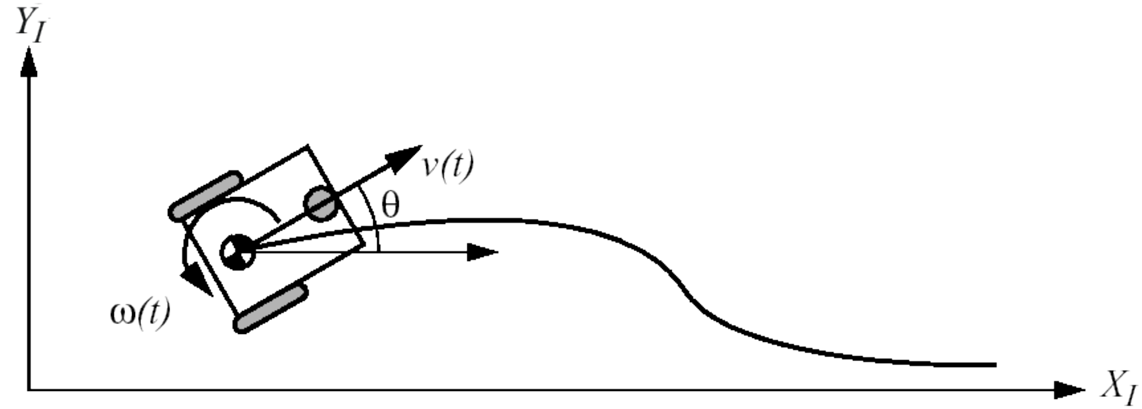
$$\begin{cases} \frac{\Delta x}{\Delta t} = \frac{v_r+v_l}{2}\cos\tilde{\theta} \\ \frac{\Delta y}{\Delta t} = \frac{v_r+v_l}{2}\sin\tilde{\theta} \\ \frac{\Delta\theta}{\Delta t} = \frac{v_r - v_l}{b} \end{cases}$$

$$\begin{cases} \Delta x = \frac{v_r\Delta t+v_l\Delta t}{2}\cos\tilde{\theta} = \frac{\Delta s_r+\Delta s_l}{2}\cos\tilde{\theta} \\ \Delta y = \frac{v_r\Delta t+v_l\Delta t}{2}\sin\tilde{\theta} = \frac{\Delta s_r+\Delta s_l}{2}\sin\tilde{\theta} \\ \Delta\theta = \frac{v_r\Delta t-v_l\Delta t}{b} = \frac{\Delta s_r-\Delta s_l}{b} \end{cases}$$

Pose Variation During Δt

$$\Delta s = \frac{\Delta s_r + \Delta s_l}{2}$$

$$\begin{cases} \Delta x = \Delta s \cos(\theta + \frac{\Delta\theta}{2}) \\ \Delta y = \Delta s \sin(\theta + \frac{\Delta\theta}{2}) \\ \Delta\theta = \frac{\Delta s_r - \Delta s_l}{b} \end{cases}$$



b = inter-wheel distance

Δs_r = traveled distance right wheel

Δs_l = traveled distance left wheel

Δθ = orientation change of the vehicle

$$p = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \xrightarrow{t'=t+\Delta t} p' = \begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix}$$

$$p' = f(x, y, \theta, \Delta s_r, \Delta s_l, b) = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} \Delta s \cos(\theta + \Delta\theta/2) \\ \Delta s \sin(\theta + \Delta\theta/2) \\ \Delta\theta \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} \frac{\Delta s_r + \Delta s_l}{2} \cos(\theta + \frac{\Delta s_r - \Delta s_l}{2b}) \\ \frac{\Delta s_r + \Delta s_l}{2} \sin(\theta + \frac{\Delta s_r - \Delta s_l}{2b}) \\ \frac{\Delta s_r - \Delta s_l}{b} \end{bmatrix}$$

$$p' = \begin{bmatrix} f_1(x, y, \theta, \Delta s_r, \Delta s_l, b) \\ f_2(x, y, \theta, \Delta s_r, \Delta s_l, b) \\ f_3(x, y, \theta, \Delta s_r, \Delta s_l, b) \end{bmatrix}$$

Non-Deterministic Uncertainities in Wheel-Based Odometry

Nondeterministic Error Sources

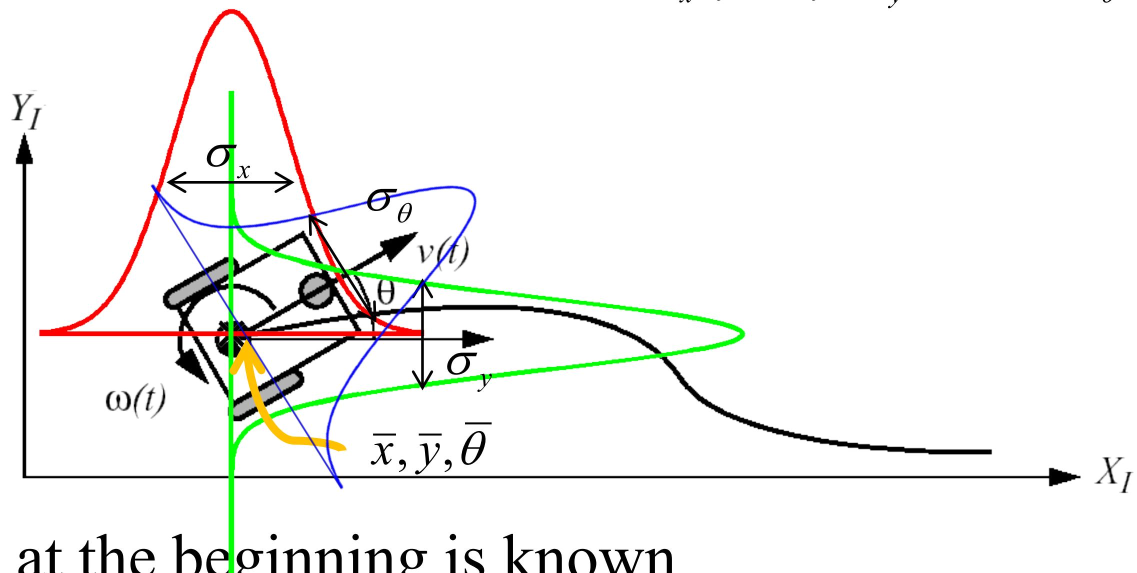
- Variation of the contact point of the wheel
- Unequal floor contact (e.g., wheel slip, nonplanar surface)

- Wheels cannot be assumed to roll perfectly
- Measured encoder values do not perfectly reflect the actual motion
- Pose error is cumulative and incrementally increases
- Probabilistic modeling for assessing quantitatively the error

Noise modeling

Model error in each dimension with a Gaussian $x \rightarrow \bar{x}, \sigma_x; y \rightarrow \bar{y}, \sigma_y; \theta \rightarrow \bar{\theta}, \sigma_\theta$

$$\Sigma_p = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 & \sigma_{x\theta}^2 \\ \sigma_{yx}^2 & \sigma_{yy}^2 & \sigma_{y\theta}^2 \\ \sigma_{\theta x}^2 & \sigma_{\theta y}^2 & \sigma_{\theta\theta}^2 \end{bmatrix}$$



Assumptions:

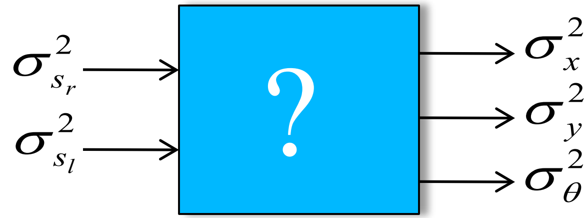
- Covariance matrix Σ_p at the beginning is known
- Errors of the two individual wheels are independent
- Errors are independent of direction of motion
- Errors are proportional to the distance traveled (k_r, k_l model parameters)

$$\sum_\Delta = \mathrm{cov}(\Delta s_r, \Delta s_l) = \begin{bmatrix} k_r|\Delta s_r| & 0 \\ 0 & k_l|\Delta s_l| \end{bmatrix} = \begin{bmatrix} \sigma_{s_r}^2 & 0 \\ 0 & \sigma_{s_l}^2 \end{bmatrix}$$

Actuator Noise → Pose Noise

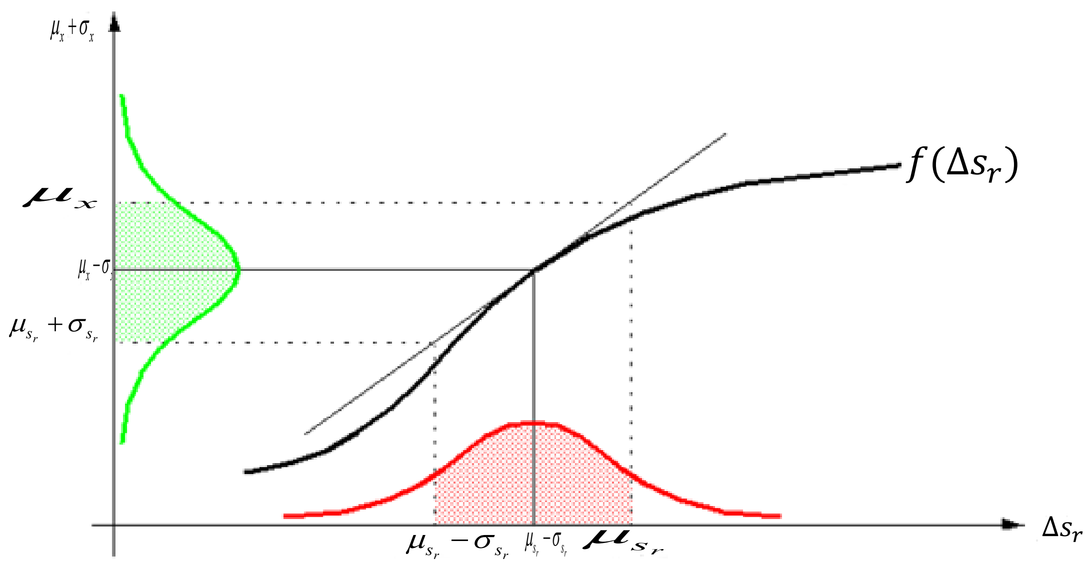
- How is the actuator noise (2D) propagated to the pose (3D)?

$$\Sigma_\Delta = \begin{bmatrix} \sigma_{s_r}^2 & 0 \\ 0 & \sigma_{s_l}^2 \end{bmatrix}$$



$$\begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 & \sigma_{x\theta}^2 \\ \sigma_{yx}^2 & \sigma_{yy}^2 & \sigma_{y\theta}^2 \\ \sigma_{\theta x}^2 & \sigma_{\theta y}^2 & \sigma_{\theta\theta}^2 \end{bmatrix} = \Sigma_p$$

- 1D to 1D example $N(\mu_{s_r}, \sigma_{s_r}) \rightarrow N(\mu_x, \sigma_x)$



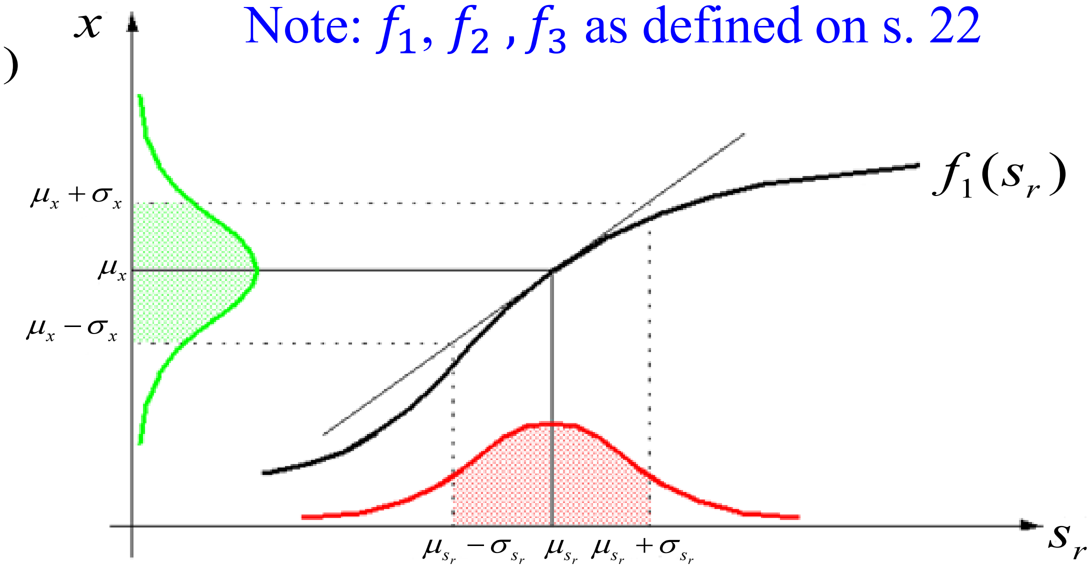
- We need to linearize → Taylor Series

$$x \approx f(\Delta s_r)\Big|_{\Delta s_r = \mu_{s_r}} \approx f(\Delta s_r) + \frac{1}{1!}\frac{\partial f}{\partial \Delta s_r}(\Delta s_r - \mu_{s_r}) + \frac{1}{2!}\frac{\partial^2 f}{\partial {\Delta s_r}^2}(\Delta s_r - \mu_{s_r})^2 + \cdots$$

Actuator Noise → Pose Noise

$$x \approx f_1(s_r)\big|_{s_r=\mu_{s_r}} \approx f_1(s_r) + \frac{\partial f_1}{\partial s_r}(s_r - \mu_{s_r})$$

Note: f_1, f_2 , f_3 as defined on s. 22

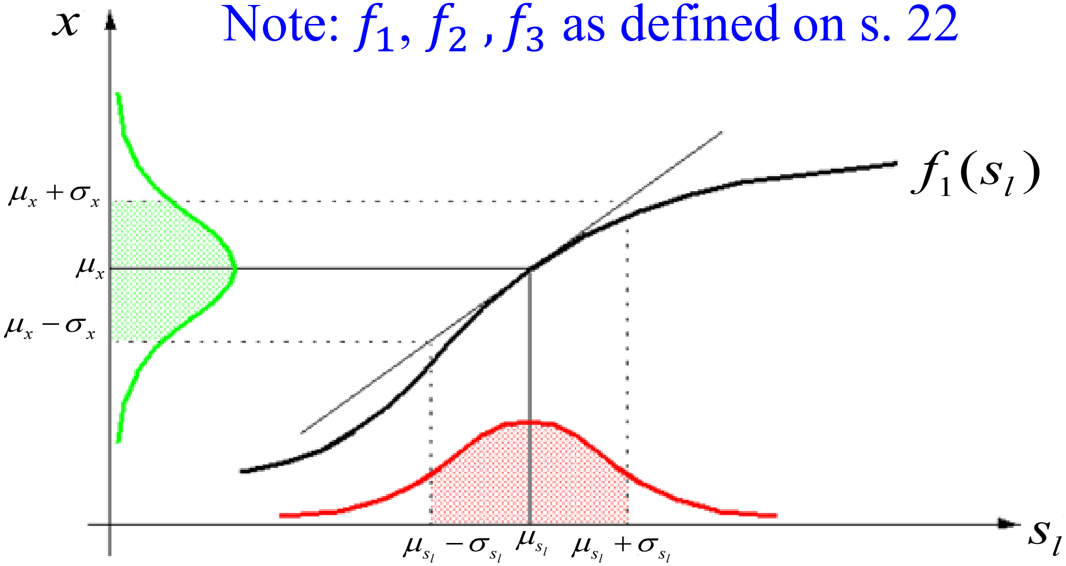


Actuator Noise → Pose Noise

$$x \approx f_1(s_r)\big|_{s_r=\mu_{s_r}} \approx f_1(s_r) + \frac{\partial f_1}{\partial s_r}(s_r - \mu_{s_r})$$

$$x \approx f_1(s_l)\big|_{s_l=\mu_{s_l}} \approx f_1(s_l) + \frac{\partial f_1}{\partial s_l}(s_l - \mu_{s_l})$$

Note: f_1, f_2 , f_3 as defined on s. 22



Actuator Noise → Pose Noise

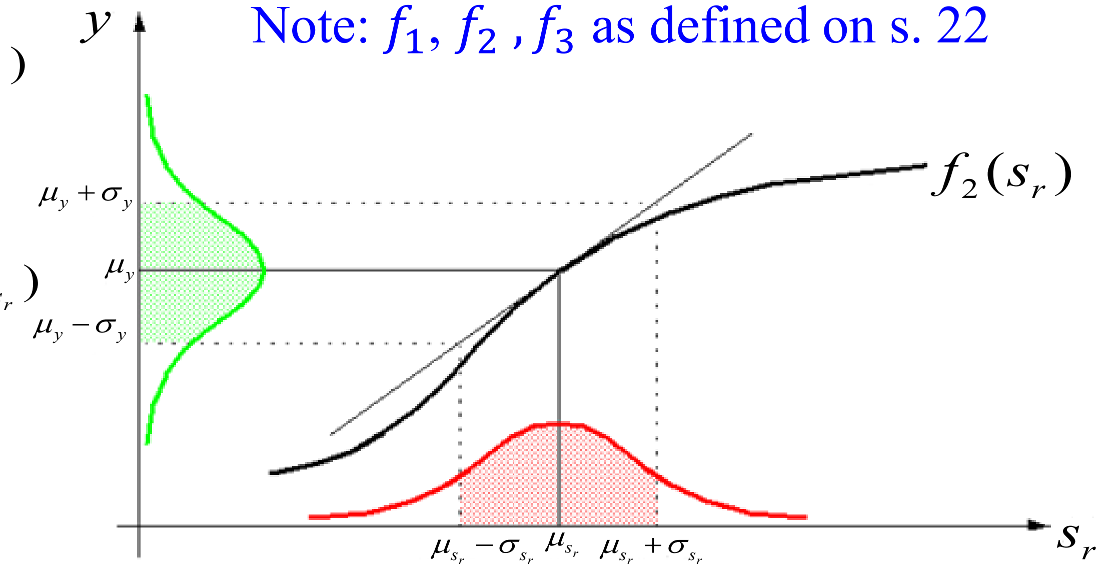
$$x \approx f_1(s_r)\big|_{s_r=\mu_{s_r}} \approx f_1(s_r) + \frac{\partial f_1}{\partial s_r}(s_r - \mu_{s_r})$$

$$x \approx f_1(s_l)\big|_{s_l=\mu_{s_l}} \approx f_1(s_l) + \frac{\partial f_1}{\partial s_l}(s_l - \mu_{s_l})$$

$$y \approx f_2(s_r)\big|_{s_r=\mu_{s_r}} \approx f_2(s_r) + \frac{\partial f_2}{\partial s_r}(s_r - \mu_{s_r})$$

$$\dots$$

Note: f_1, f_2 , f_3 as defined on s. 22



$$F_{\Delta rl} = \begin{bmatrix} \frac{\partial f_1}{\partial \Delta s_r} & \frac{\partial f_1}{\partial \Delta s_l} \\ \frac{\partial f_2}{\partial \Delta s_r} & \frac{\partial f_2}{\partial \Delta s_l} \\ \frac{\partial f_3}{\partial \Delta s_r} & \frac{\partial f_3}{\partial \Delta s_l} \end{bmatrix}$$ Jacobian

- General error propagation law

$$\Sigma_{\Delta rl} = F_{\Delta rl} \Sigma_{\Delta} F_{\Delta rl}^T \qquad \text{with } \Sigma_{\Delta} = \begin{bmatrix} k_r|\Delta s_r| & 0 \\ 0 & k_l|\Delta s_l| \end{bmatrix}$$

Actuator Noise → Pose Noise

How does the pose covariance Σ_p evolve over time?

- Initial covariance of vehicle at t=0:

$$\Sigma_p^{(t=0)} = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 & \sigma_{x\theta}^2 \\ \sigma_{yx}^2 & \sigma_{yy}^2 & \sigma_{y\theta}^2 \\ \sigma_{\theta x}^2 & \sigma_{\theta y}^2 & \sigma_{\theta\theta}^2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

- Additional noise at each time step Δt: $\Sigma_{\Delta rl} = F_{\Delta rl} \Sigma_\Delta F_{\Delta rl}^T$
- Covariance at t=1Δt: $\Sigma_p^{(t=1\Delta t)} = \Sigma_p^{(t=0)} + \Sigma_{\Delta rl} = \Sigma_{\Delta rl}$
- Covariance at t=2Δt:

$$\Sigma_p^{(t=2\Delta t)} = F_p \Sigma_p^{(t=1\Delta t)} F_p^T + F_{\Delta rl} \Sigma_\Delta F_{\Delta rl}^T$$

$$F_p = \begin{bmatrix} \frac{\partial f_1}{\partial_x} & \frac{\partial f_1}{\partial_y} & \frac{\partial f_1}{\partial_\theta} \\ \frac{\partial f_2}{\partial_x} & \frac{\partial f_2}{\partial_y} & \frac{\partial f_2}{\partial_\theta} \\ \frac{\partial f_3}{\partial_x} & \frac{\partial f_3}{\partial_y} & \frac{\partial f_3}{\partial_\theta} \end{bmatrix}$$

Note: f_1, f_2 , f_3 as defined on s. 22

Actuator Noise → Pose Noise

Algorithm

Precompute:

- Determine actuator noise Σ_Δ
- Compute mapping actuator-to-pose noise incremental $F_{\Delta rl}$
- Compute mapping pose propagation noise over step F_p

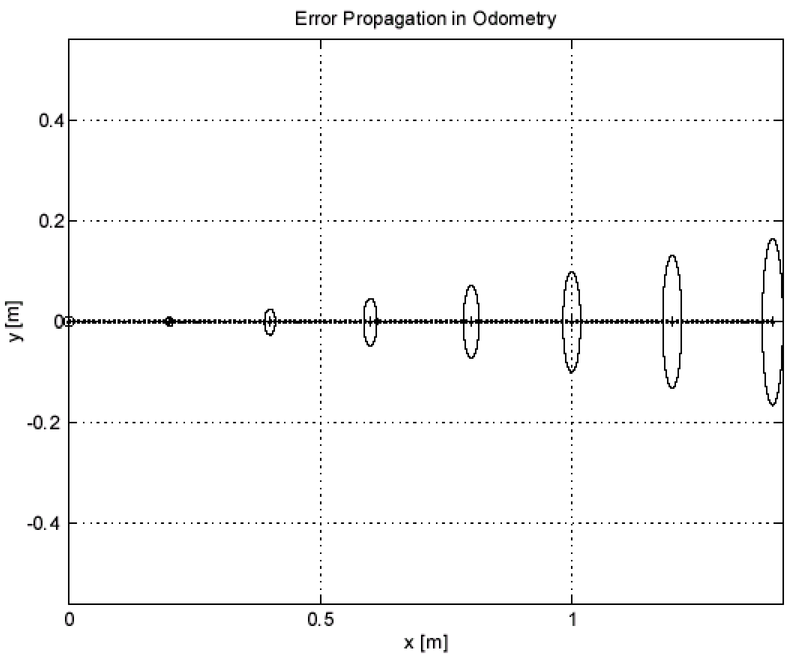
Initialize:

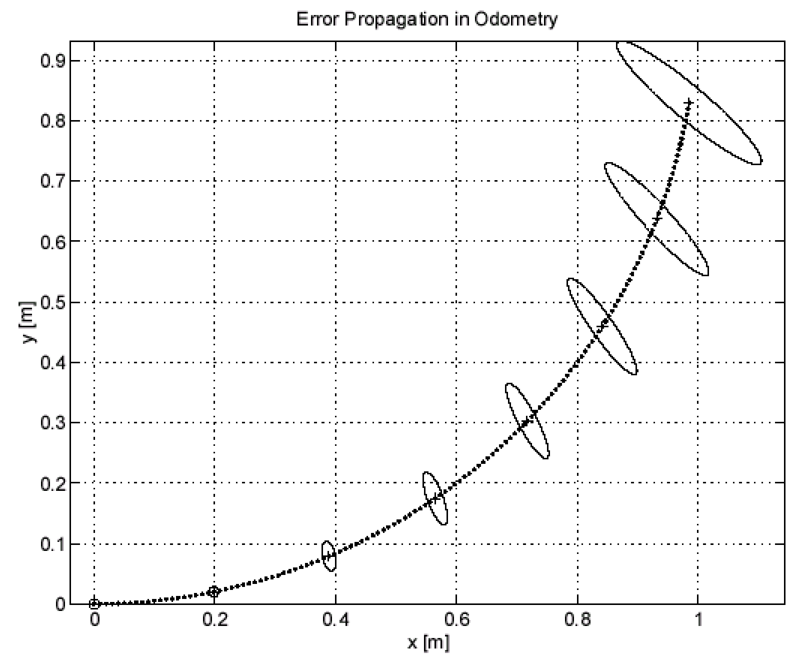
- Initialize $\Sigma_p^{(t=0)} = [0]$

Iterate:

$$\Sigma_p^{(t=(k+1)\Delta t)} = F_p \Sigma_p^{(t=k\Delta t)} F_p^T + F_{\Delta rl} \Sigma_\Delta F_{\Delta rl}^T$$

Classical 2D Representation



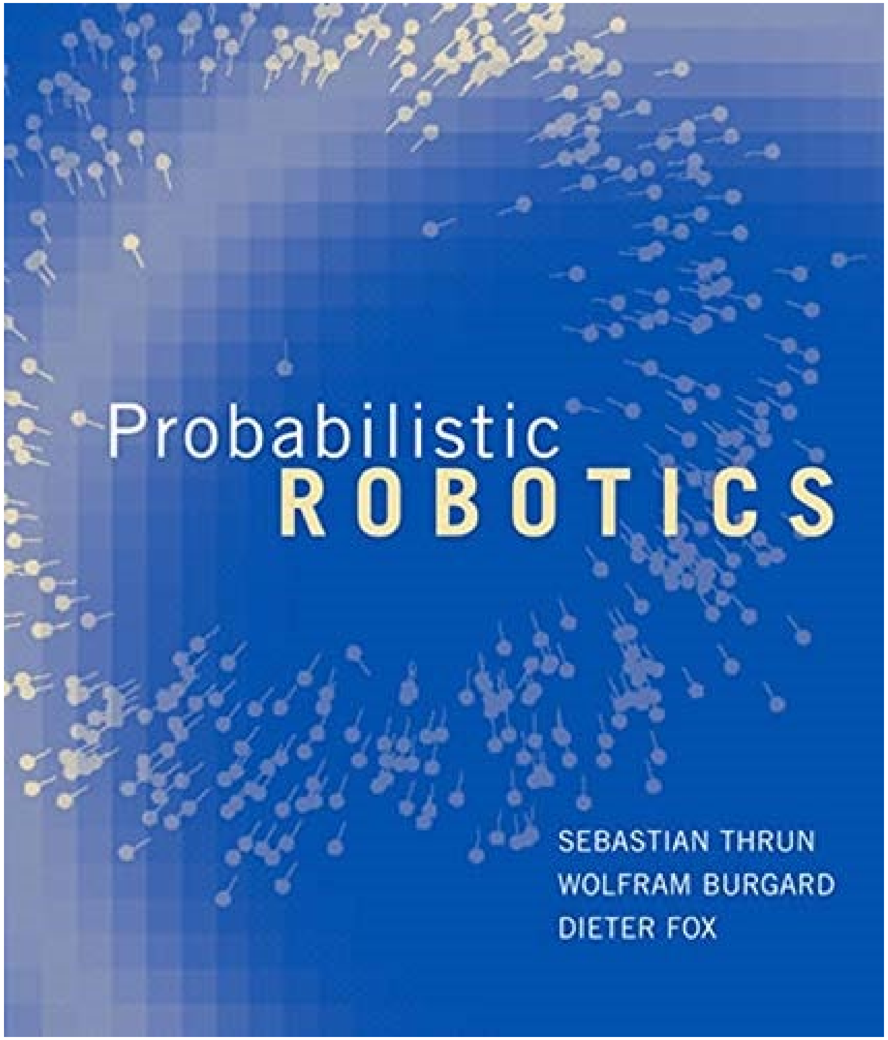


Courtesy of R. Siegwart and R. Nourbakhsh

Ellipses: typical 3σ bounds

The Multi-Dimensional Kalman Filter Algorithm and its Application to Localization

Two Key Sources of Information



Multi-Dimensional Kalman Filter

Estimates the state x of a discrete-time controlled process that is governed by the linear stochastic difference equation

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t$$

with a measurement

$$z_t = C_t x_t + \delta_t$$

x : state vector, dim $(x) =$ n
u : control vector, dim $(u) = m$
z : measurement vector dim $(u) = k$

[Adapted from Thrun et al., 2005]

Components of a Kalman Filter

A_t Matrix (nxn) that describes how the state evolves from t to $t\text{-}1$ without controls or noise.

B_t Matrix (nxm) that describes how the control u_t changes the state from t to $t\text{-}1$.

C_t Matrix (kxn) that describes how to map the state x_t to an observation z_t.

ε_t

δ_t

Random variables representing the process and measurement noise that are assumed to be independent and normally distributed with covariance R_t and Q_t respectively.

[Adapted from Thrun et al., 2005]

Kalman Filter Algorithm

1. Algorithm **Kalman_filter**(μ_{t-1}, Σ_{t-1}, u_t, z_t)

Prediction:

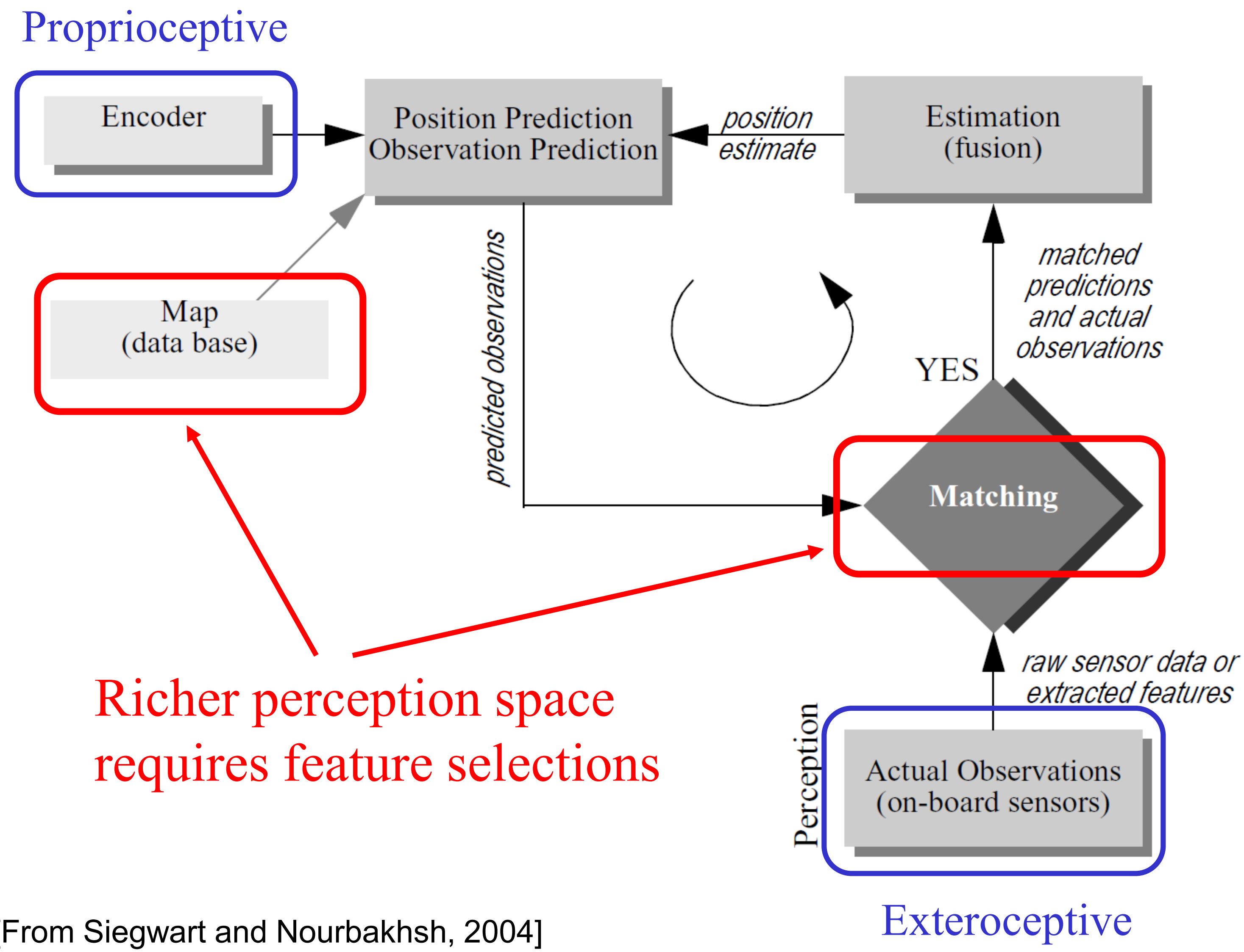
2. $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$
3. $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$

Correction or update:

4. $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$
5. $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$
6. $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$
7. Return μ_t, Σ_t

[Adapted from Thrun et al., 2005]

2D Localization with Kalman Filter



[From Siegwart and Nourbakhsh, 2004]

Conclusion on Localization

Take Home Messages

- Feature-based localization is a way to compensate odometry limitations by leveraging exteroceptive sensors in addition to proprioceptive ones
- A Kalman filter is a computationally efficient, optimal recursive data processing algorithm that allows fusion of multiple estimates coming from either process or sensing models
- A Kalman filter assumes linear motion and sensor models characterized by white Gaussian noise: not all problems in robot localization fulfill these assumptions; other computationally more expensive techniques (e.g., Particle Filters) are available for such problems
- The impact of non-deterministic error sources affecting motion at the actuator level (e.g., wheel-ground interaction) can be modeled probabilistically and forward-propagated to the pose thanks to a kinematic forward model of the vehicle

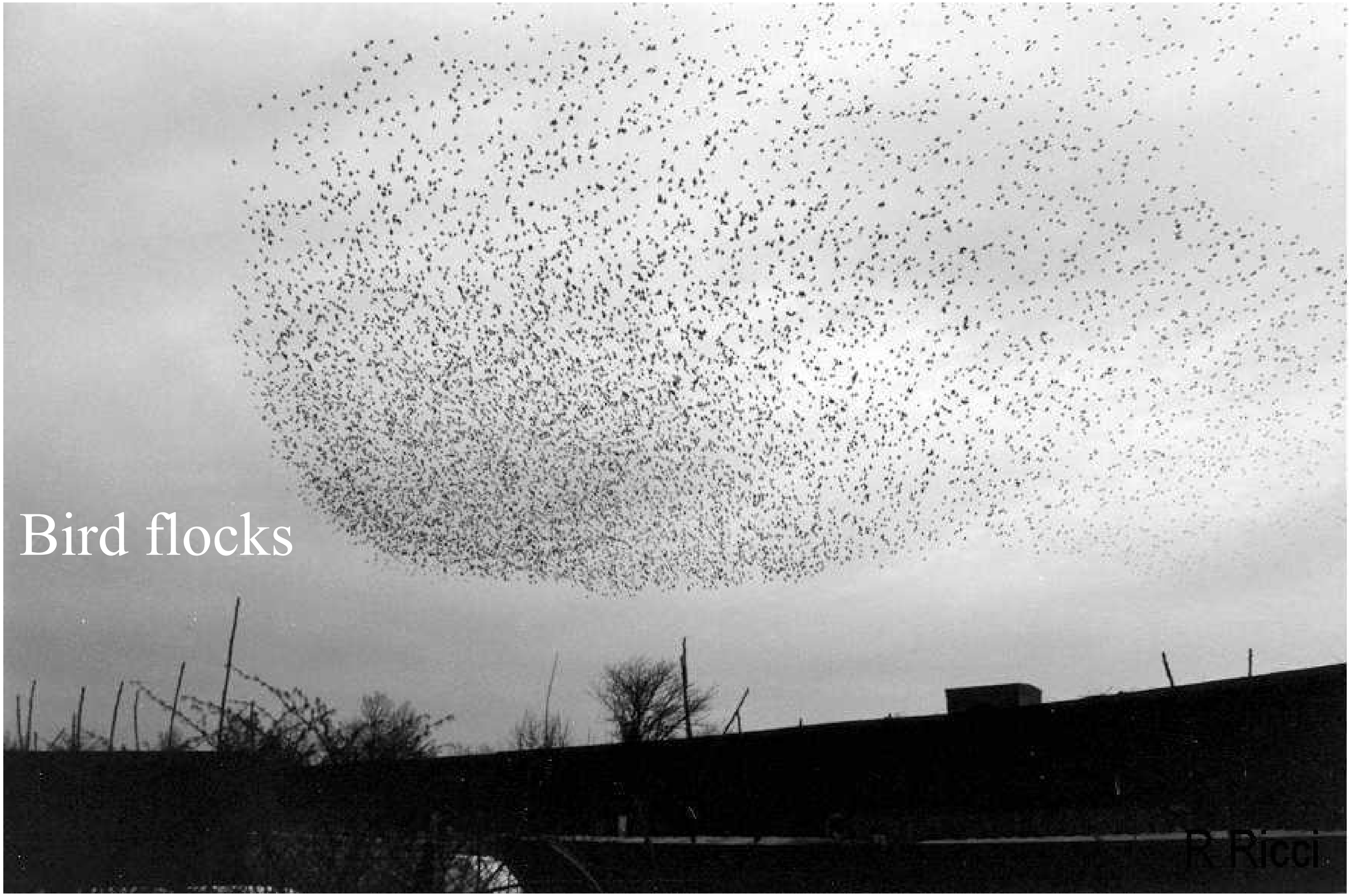
Collective Movements in Natural Societies: Phenomena


上海市第一百货商店
OMEGA

43


Fish schools


Insect swarms


Bird flocks
R Ricci

Bird flocks

Flocking in Animal Societies

Seems to occur in

- All media (air, water, land)
- Many animal families (insects, fish, birds, mammals...)
- From small groups (2 geese) to enormous groups (herring shoals 17 miles long)
- Animals of different ages and sizes
- In some animals, only in special circumstances (e.g. migration)

Flocking Phenomena

Rapid directed movement of the whole flock

Reactivity to predators (flash expansion, fountain effect)

Reactivity to obstacles

No collisions between flock members

Coalescing and splitting of flocks

Tolerant of movement within the flock, loss or gain of flock members

No dedicated leader

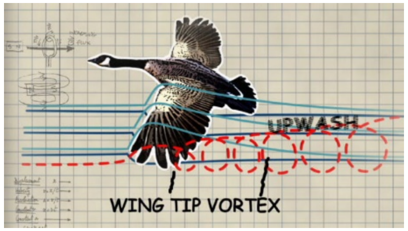
Different species can have different flocking characteristics – easy to recognize but not always easy to describe

Benefits of Flocking: 1- Energy saving

Example - V-formations in birds:

- Geese flying in Vs can extend their flight range by over 70%
- Birds in flocks generally fly faster than when flying alone

Reason: Each bird rides on the vortex cast off by the wing-tip of the one in front (i.e., slightly above and towards either side of the bird in front)



- Cyclists save energy in similar way.

Benefits of Flocking: 2- Navigation Accuracy

Several examples:

- Monarch butterflies reach the same trees every year
- Wrynecks (migratory woodpecker) do the same from Africa to Valais
- Fish reach the same tiny spawning grounds (i.e., egg deposition)

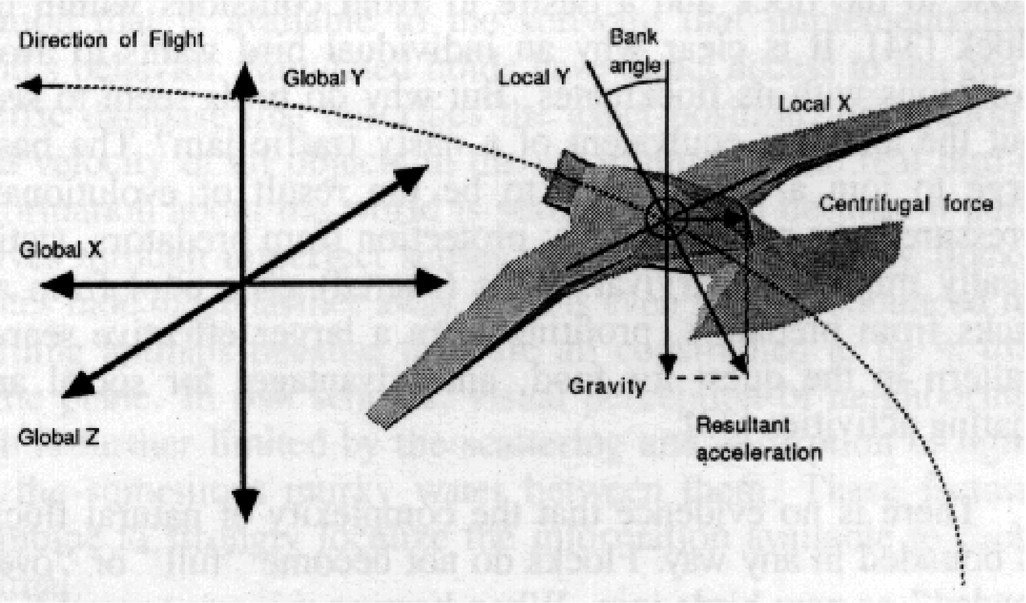
Flocking in Simple Virtual Agents: Reynolds' Boids

Craig Reynolds' Boids (1987)

A computer animator who wanted to find a way of animating flocks that would be

- Realistic looking
- Computationally efficient, with complexity preferably no worse than linear in number of flockmates -> actually obtained in 1987 $O(n^2)$
- 3D

Boids' Flight Model



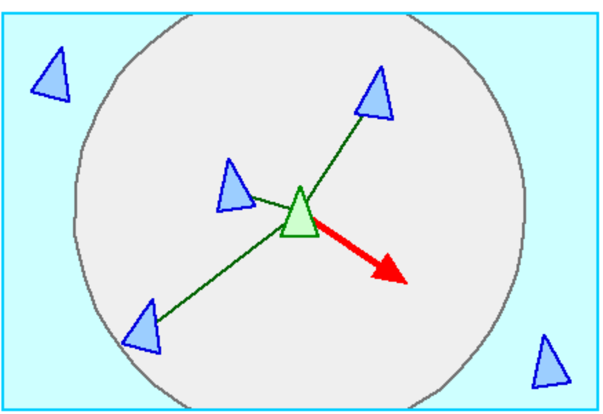
A simple 3D model:

- orientation
- momentum conservation
- maximal acceleration
- maximal speed via viscous friction
- some gravity + aerodynamic lift (slow ramp up, fast ramp down and stall possible)
- wings flapping independently, just for making it more realistic

Reynolds’ Rules for Flocking

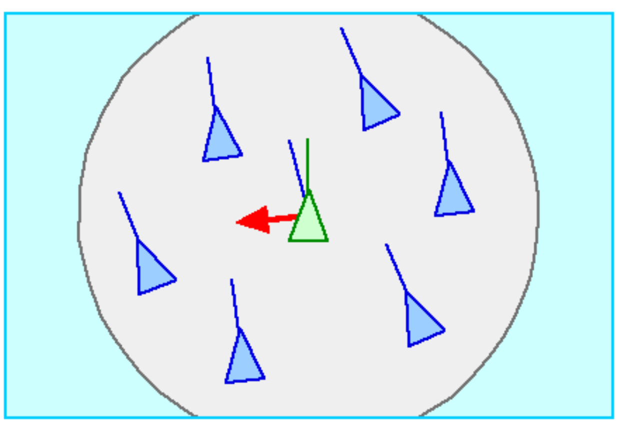
1. Separation: avoid collisions with nearby flockmates
2. Alignment: attempt to match velocity (speed and direction) with nearby flockmates
3. Cohesion: attempt to stay close to nearby flockmates

Position control



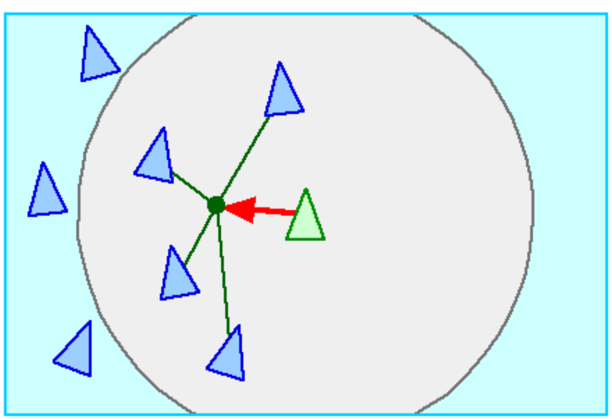
separation

Velocity control



alignment

Position control



cohesion

Arbitrating Rules

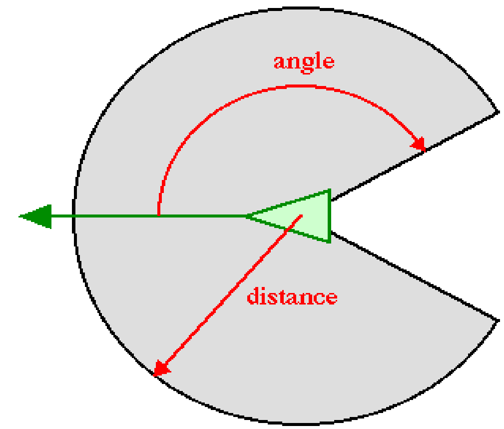
- Boids' controller is rule-based (or behavior-based)
- Time-constant linear weighted sum did not work in front of obstacles
- Time-varying, nonlinear weighted sum worked much better: allocate the maximal acceleration available to the highest behavioral priority, the remaining to the other behaviors
- Separation > alignment > cohesion → splitting possible in front of an obstacle

Great example of mixing principles behind Arkin's motor schemas and Brooks' subsumption architectures!

Sensory System for Teammate Detection

An idealized system (but distributed and local!):

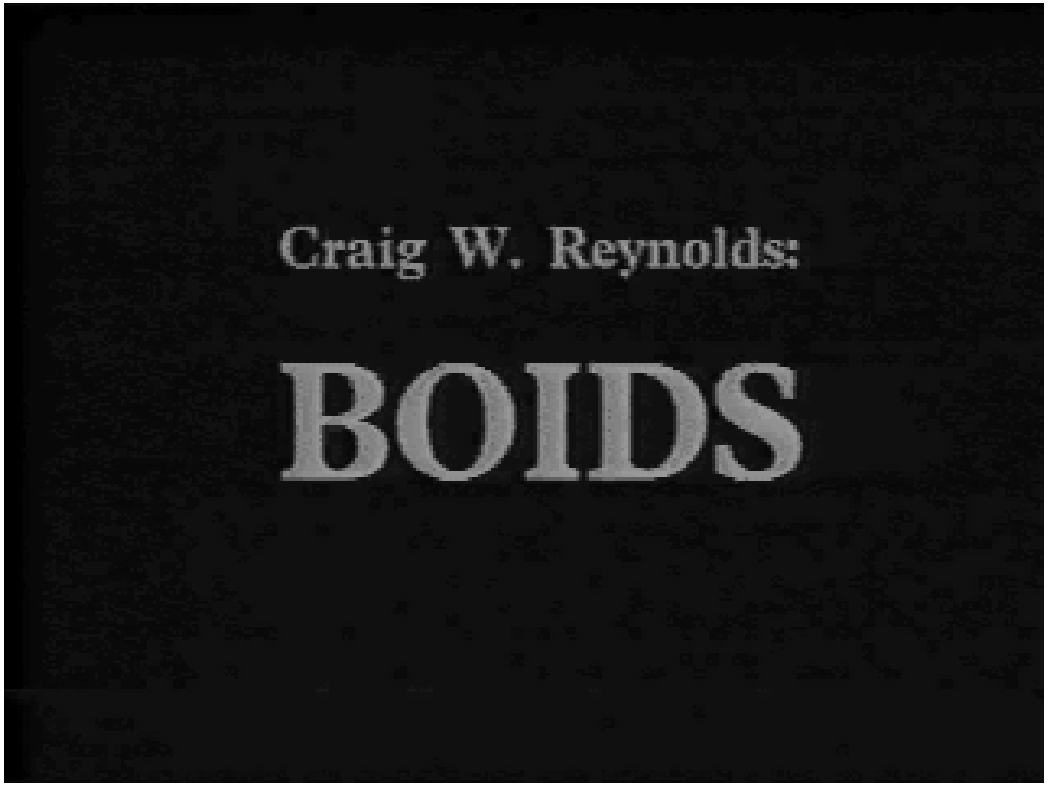
- Local, almost omni-directional sensory system
- Perfect relative range and bearing system: no occlusion, no noise, all teammates perfectly identified within the range of detection
- Immediate response: one perception-to-action loop (no sensory, computational capacity considered)
- Homogeneous system (all Boids have exactly the same sensory system)
- "Natural" nonlinearities: negative exponential of the distance (linear response also tested: bouncy, cartoony)



Neighborhood (2D version)

Flocking without Obstacles

Does it work? Does it produce realistic flocking?
Judge for yourselves.



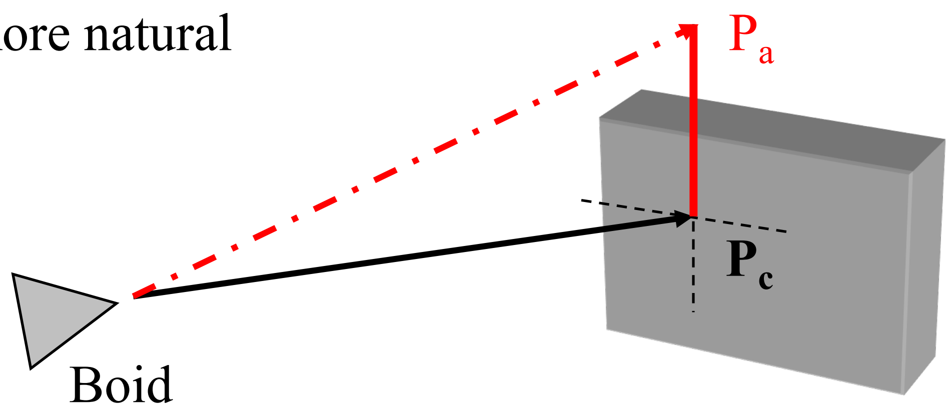
Moving from A to B

The migratory urge

- Reynolds wanted to be able to direct the flocks along particular courses and to program scripted movements
- He added a low priority acceleration request (the migratory urge) towards a point or in a direction
- By moving the target point, he could steer the flock around the environment
- Discrete jumps in the position of the point resulted in smooth changes of direction

Dealing with Obstacles

- Sensory system for environmental obstacle detection: different from that used to perceive teammates in Boids!
- Approach 1: potential fields
  - Repulsive force field around the obstacle
  - See week 3 lecture, Arkin's motor schemas
  - Poor results in Boids
- Approach 2: steer-to-avoid
  - Consider obstacles ONLY directly in the front
  - Find the silhouette edge closest to the point of collision (P_c)
  - Aim the Boid one body length outside that edge (P_a)
  - Worked much better; also more natural



Flocking with Obstacles

Does it work? Does it produce realistic flocking?

Judge for yourselves.

What Happens if You Mess Around

Ex. omit alignment (velocity matching):

- flocking happens
- but the flocks aren't intrinsically polarized
- but you can polarize them with a strong migratory urge
- flocks look like swarms of flies

Note: all real robot experiments up to date have mainly focused on implementing exclusively rule 1 (separation) and 3 (cohesion)! Not really polarized flocks ….

More on Boids …

Craig Reynolds' web page on Boids

http://www.red3d.com/cwr/boids/

- lots of links
- lots of downloadable code (including source code)
- lots of references

Conclusion on Collective Movements

Take Home Messages

- Flocking and shoaling phenomena in vertebrates are self-organized structures emerging from local rules
- Major breakthrough through Reynold's work and his three rules – separation, alignment, cohesion
- The three rules alone are not enough for realistic scenarios: obstacle avoidance and migratory urge are part of the solution
- Major differences between virtual and real agents in communication, sensing, actuation, and control: for instance, most of flocking with real robots did not use the alignment rule also because sensing velocity of teammates is difficult

Additional Literature – Week 5

Pointers

http://rossum.sourceforge.net/papers/DiffSteer/

http://rossum.sourceforge.net/tools/MotionApplet/MotionApplet.html

http://www.probabilistic-robotics.org/

Books

- Siegwart R., Nourbakhsh I., and Scaramuzza D., "Introduction to Autonomous Mobile Robots, second Edition", MIT Press, 2011.
- Thrun S., Burgard W., and Fox D., Probabilistic Robotics, MIT Press, 2005.
- Choset H., Lynch K. M., Hutchinson S., Kantor G., Burgard W., Kavraki L., and Thrun S., "Principles of Robot Motion". MIT Press, 2005.
- Borenstein J., Everett H. R., and Feng L. "Navigating Mobile Robots: Systems and Techniques", A. K. Peters, Ltd., 1996.
- Parrish J. K. and Hamner W. M., "Animal Groups in Three Dimensions". Cambridge University Press, 1997.
- Krause J. and Ruxton G. D. "Living in groups", Oxford University Press, 2002.