Lecture 19
Mobile Robotics - III - Kalman

Rudolf E. Kálmán

文A 31 languages

Article Talk Read Edit View history Tools

From Wikipedia, the free encyclopedia

The native form of this personal name is Kálmán Rudolf Emil. *This article uses Western name order when mentioning individuals.*

Rudolf Emil Kálmán[3] (May 19, 1930 – July 2, 2016) was a Hungarian-American electrical engineer, mathematician, and inventor. He is most noted for his co-invention and development of the Kalman filter, a mathematical algorithm that is widely used in signal processing, control systems, and guidance, navigation and control. For this work, U.S. President Barack Obama awarded Kálmán the National Medal of Science on October 7, 2009.[4]

Rudolf E. Kálmán

Born	Rudolf Emil Kálmán[1] May 19, 1930 Budapest, Hungary
Died	July 2, 2016 (aged 86)[2] Gainesville, Florida
Citizenship	Hungary United States
Alma mater	Massachusetts Institute of Technology Columbia University

Life and career [edit]

Rudolf Kálmán was born in Budapest, Hungary, in 1930 to Otto and Ursula Kálmán (née Grundmann). After emigrating to the United States in 1943, he earned his bachelor's degree in 1953 and his master's degree in 1954, both from the Massachusetts Institute of Technology, in electrical engineering. Kálmán completed his doctorate in 1957 at Columbia University in New York City.[5]

Kálmán worked as a Research Mathematician at the Research Institute for Advanced Studies in Baltimore, Maryland, from 1958 until 1964. He was a professor at Stanford University from 1964 until 1971, and then a Graduate Research Professor and the Director of the Center for Mathematical System Theory, at the University of Florida from 1971 until 1992. He periodically returned to Fontainebleau from 1969 to 1972 at MINES ParisTech where he served as scientific advisor for Centre de recherches en automatique. Starting in 1973, he also held the chair of Mathematical System Theory at the Swiss Federal Institute of Technology in Zürich, Switzerland.

Kálmán died on the morning of July 2, 2016, at his home in Gainesville, Florida.[6]

from Probabilistic Robotics

Course logistics

- **Quiz 18-19 will be posted Nov 15**
- Project 4 is posted on 10/30 and will be due **11/15 (extended)**
- Details on the Final Project will be announced on 11/15.

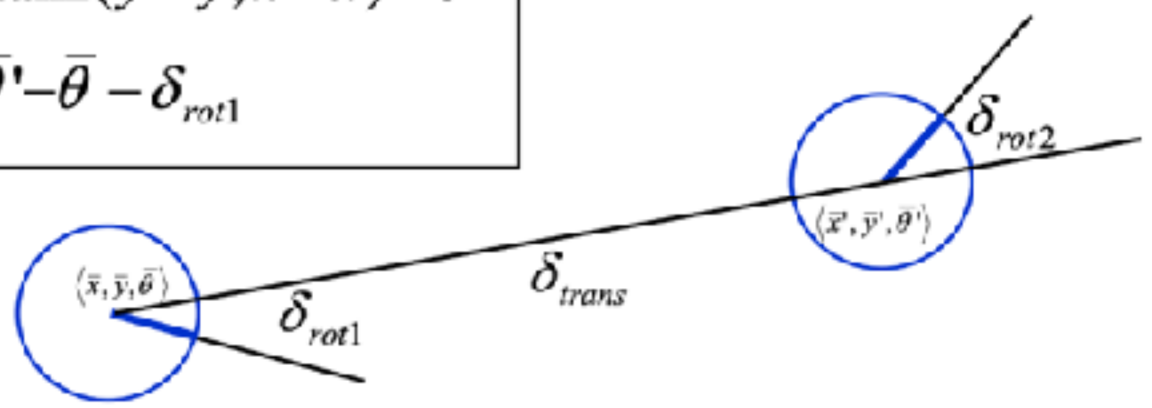
Previously

$$Bel(x_t) = \eta P(z_t \mid x_t) \int P(x_t \mid x_{t-1} u_t) Bel(x_{t-1}) dx_{t-1}$$

Probabilistic Kinematics

- Robot moves from $<\bar{x}, \bar{y}, \bar{\theta}>$ to $<\bar{x}', \bar{y}', \bar{\theta}'>$.
- Odometry information $u = <\delta_{rot1}, \delta_{trans}, \delta_{rot2}>$.

$$\delta_{trans} = \sqrt{(\bar{x}' - \bar{x})^2 + (\bar{y}' - \bar{y})^2}$$
$$\delta_{rot1} = \text{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta}$$
$$\delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1}$$



Algorithm **motion_model_odometry** (u, x, x'):

1. $\delta_{trans} = \sqrt{(\bar{x} - \bar{x}')^2 + (\bar{y} - \bar{y}')^2}$
2. $\delta_{rot1} = \text{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta}$
3. $\delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1}$

$u = \langle \delta_{rot1}, \delta_{rot2}, \delta_{trans} \rangle$

4. $\hat{\delta}_{trans} = \sqrt{(x - x')^2 + (y - y')^2}$
5. $\hat{\delta}_{rot1} = \text{atan2}(y' - y, x' - x) - \theta$
6. $\hat{\delta}_{rot2} = \theta' - \theta - \hat{\delta}_{rot1}$

$x = <x, y, \theta>$
$x' = <x', y', \theta'>$

Finding the posterior

7. $p_1 = \text{prob}(\delta_{rot1} - \hat{\delta}_{rot1}, \alpha_1 \hat{\delta}_{rot1}^2 + \alpha_2 \hat{\delta}_{trans}^2)$
8. $p_2 = \text{prob}(\delta_{trans} - \hat{\delta}_{trans}, \alpha_3 \hat{\delta}_{trans}^2 + \alpha_4(\hat{\delta}_{rot1}^2 + \hat{\delta}_{rot2}^2))$
9. $p_3 = \text{prob}(\delta_{rot2} - \hat{\delta}_{rot2}, \alpha_1 \hat{\delta}_{rot2}^2 + \alpha_2 \hat{\delta}_{trans}^2)$
10. Return $p_1 * p_2 * p_3$

Noise Model for Motion

- The measured motion is given by the true motion corrupted with noise.

$$\hat{\delta}_{rot1} = \delta_{rot1} + \varepsilon_{\alpha_1 |\delta_{rot1}| + \alpha_2 |\delta_{trans}|}$$
$$\hat{\delta}_{trans} = \delta_{trans} + \varepsilon_{\alpha_3 |\delta_{trans}| + \alpha_4 |\delta_{rot1} + \delta_{rot2}|}$$
$$\hat{\delta}_{rot2} = \delta_{rot2} + \varepsilon_{\alpha_1 |\delta_{rot2}| + \alpha_2 |\delta_{trans}|}$$

1. Algorithm **sample_motion_model** (u, x):
   $u = \langle \delta_{rot1}, \delta_{rot2}, \delta_{trans} \rangle, x = \langle x, y, \theta \rangle$
1. $\hat{\delta}_{rot1} = \delta_{rot1} + \text{sample}(\alpha_1 |\delta_{rot1}| + \alpha_2 \, \delta_{trans})$
2. $\hat{\delta}_{trans} = \delta_{trans} + \text{sample}(\alpha_3 \, \delta_{trans} + \alpha_4 (|\delta_{rot1}| + |\delta_{rot2}|))$
3. $\hat{\delta}_{rot2} = \delta_{rot2} + \text{sample}(\alpha_1 |\delta_{rot2}| + \alpha_2 \, \delta_{trans})$

4. $x' = x + \hat{\delta}_{trans} \cos(\theta + \hat{\delta}_{rot1})$
5. $y' = y + \hat{\delta}_{trans} \sin(\theta + \hat{\delta}_{rot1})$
6. $\theta' = \theta + \hat{\delta}_{rot1} + \hat{\delta}_{rot2}$

7. Return $\langle x', y', \theta' \rangle$

Beam-based Sensor Model

- Scan z consists of K measurements.

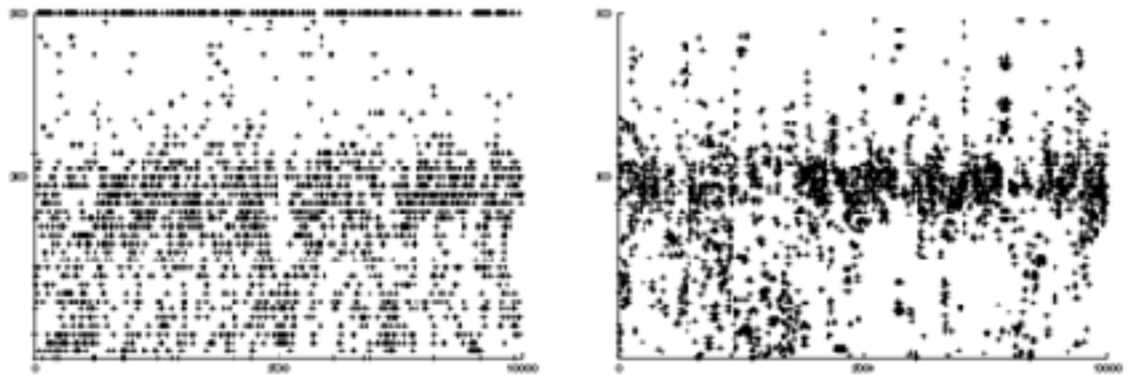
$$z = \{z_1, z_2, \ldots, z_K\}$$

- Individual measurements are independent given the robot position and a map.

$$P(z \mid x, m) = \prod_{k=1}^{K} P(z_k \mid x, m)$$

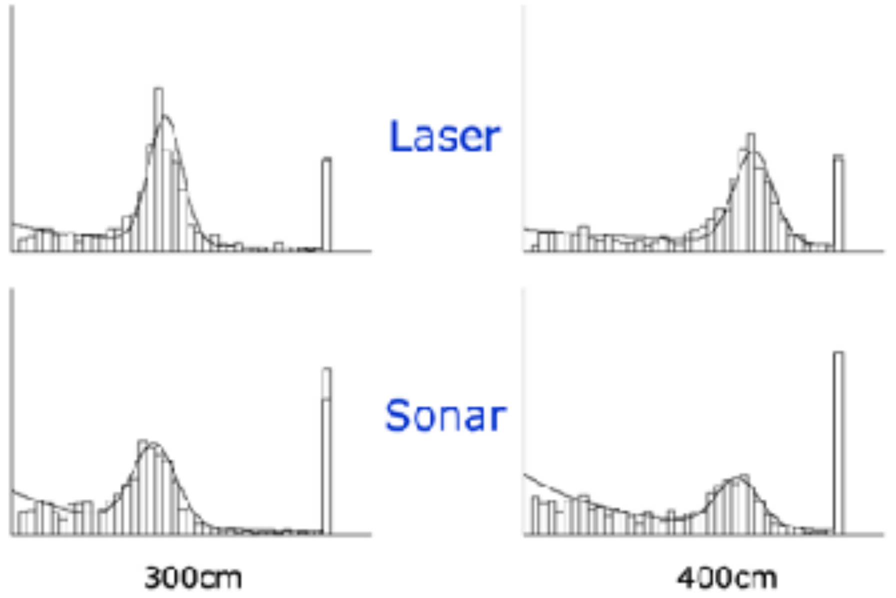
Raw Sensor Data

Measured distances for expected distance of 300 cm.

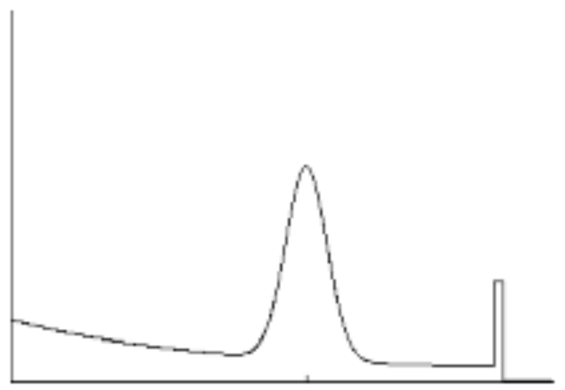


Sonar Laser

Approximation Results



Mixture Density



$$P(z \mid x, m) = \begin{pmatrix} \alpha_{\text{hit}} \\ \alpha_{\text{unexp}} \\ \alpha_{\text{max}} \\ \alpha_{\text{rand}} \end{pmatrix}^T \cdot \begin{pmatrix} P_{\text{hit}}(z \mid x, m) \\ P_{\text{unexp}}(z \mid x, m) \\ P_{\text{max}}(z \mid x, m) \\ P_{\text{rand}}(z \mid x, m) \end{pmatrix}$$

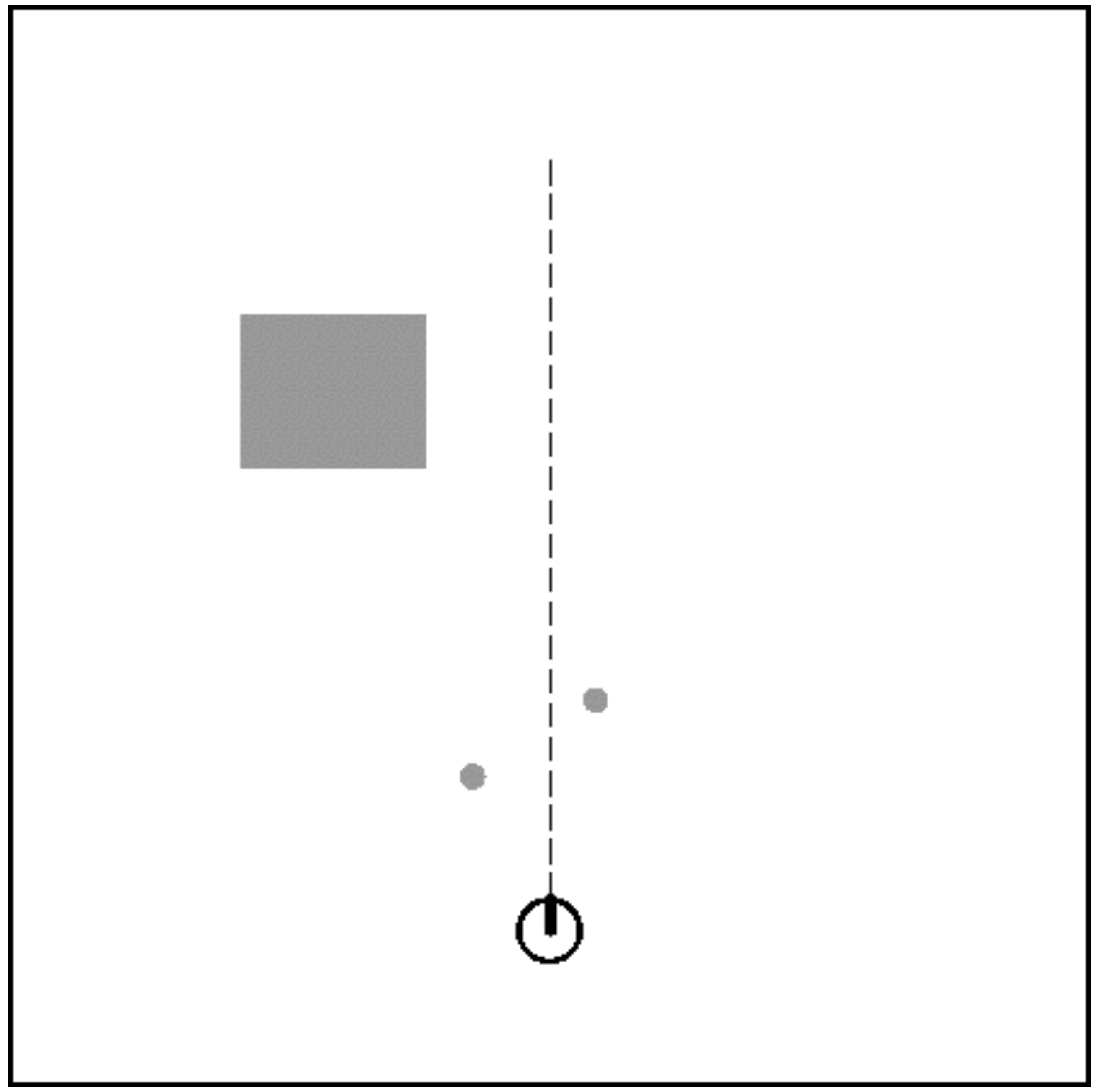
Continuing previous Lecture
Sensor Modeling

Scan-based Model

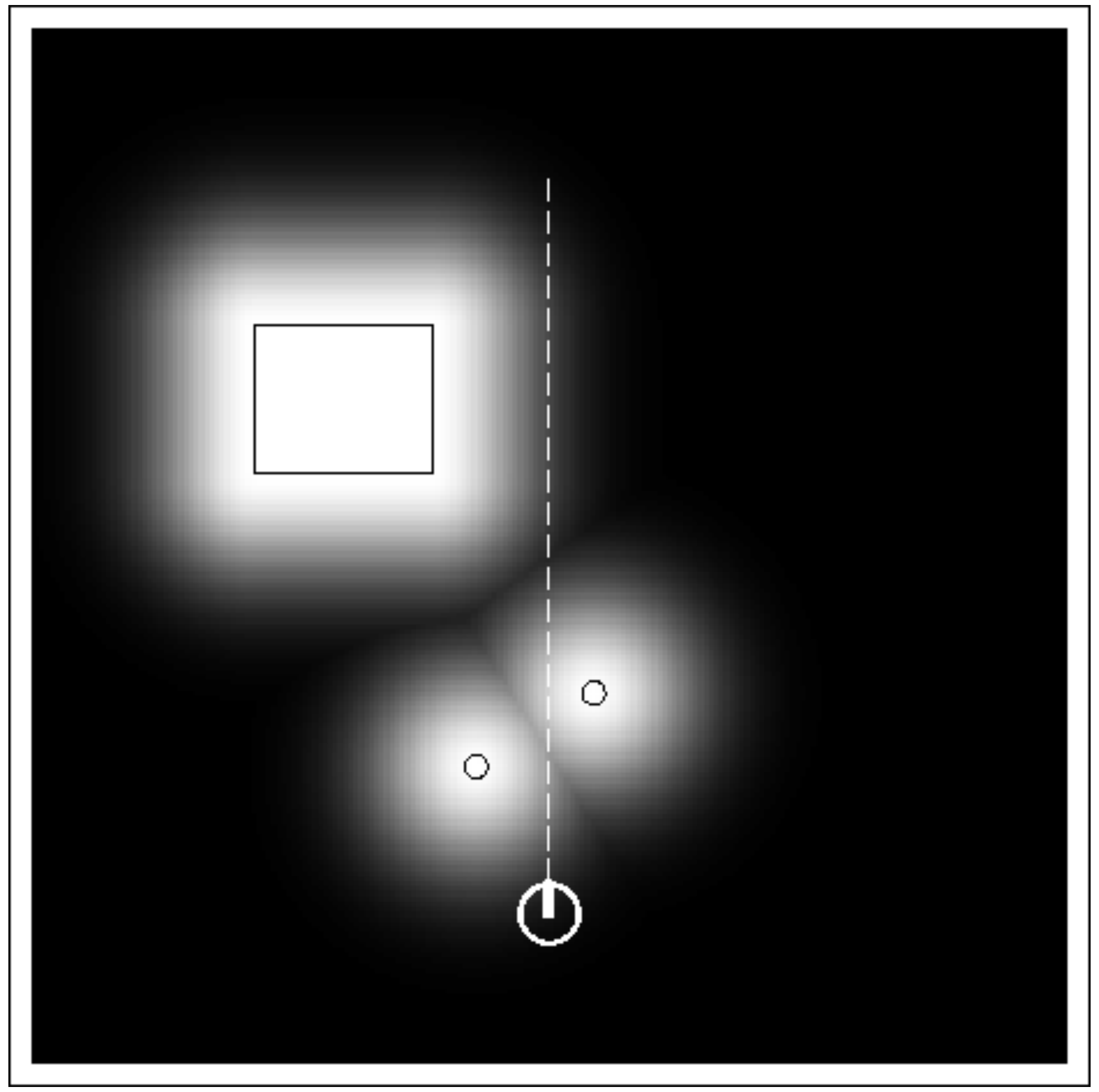
- Beam-based model is …
  - not smooth for small obstacles and at edges.
  - not very efficient.

- Idea: Instead of following along the beam, just check the end point.

Example

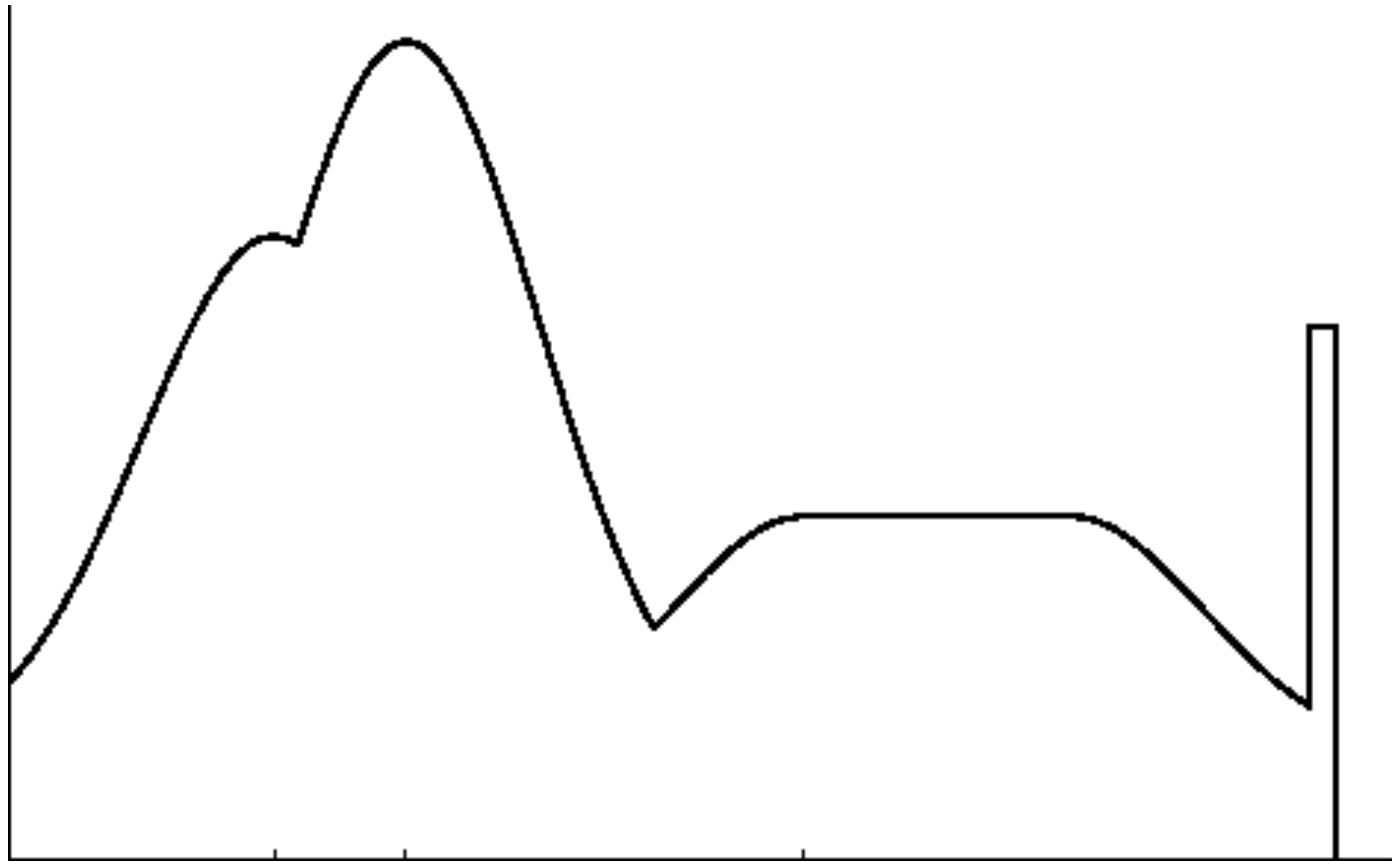


Map m

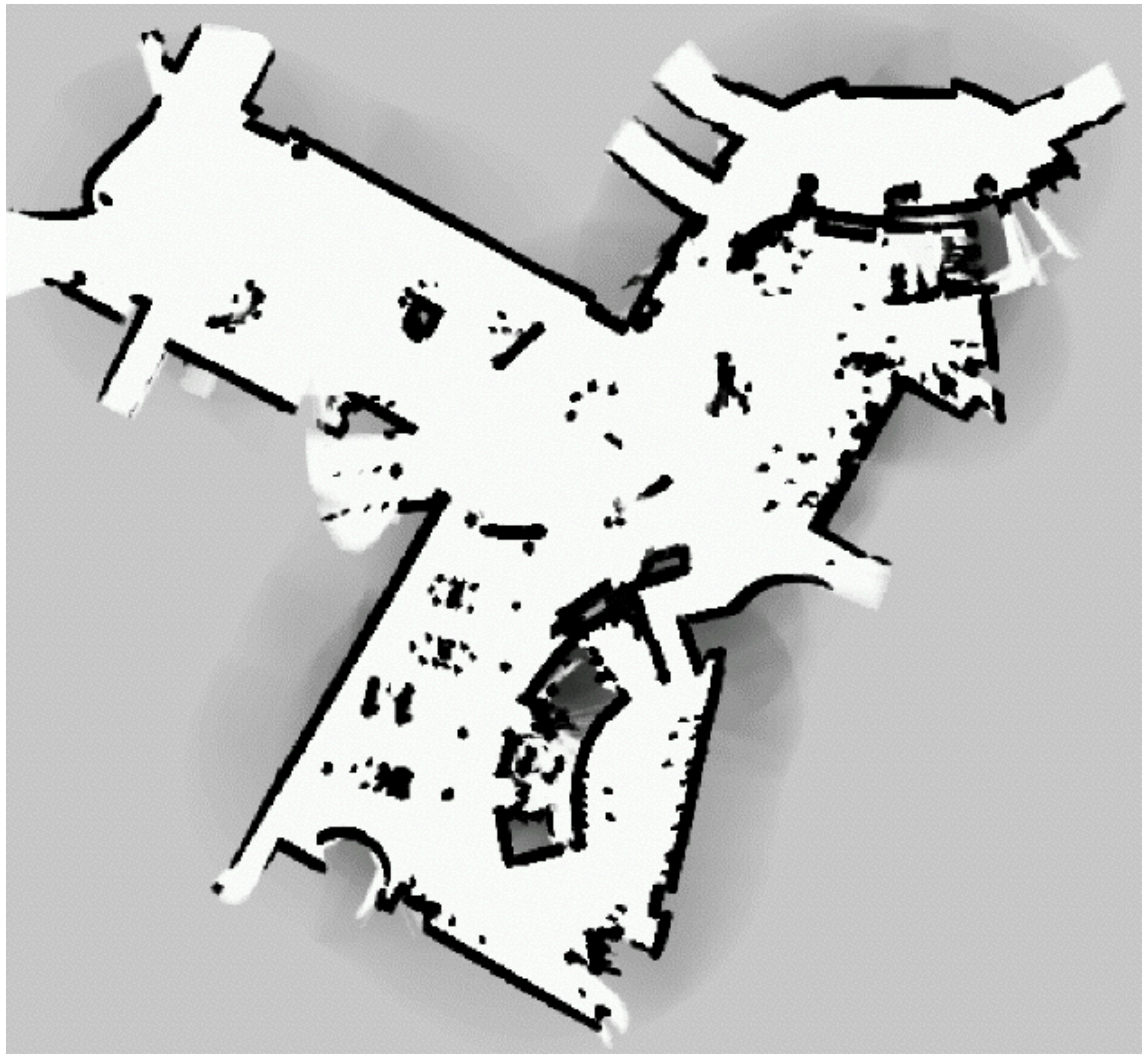


Likelihood field

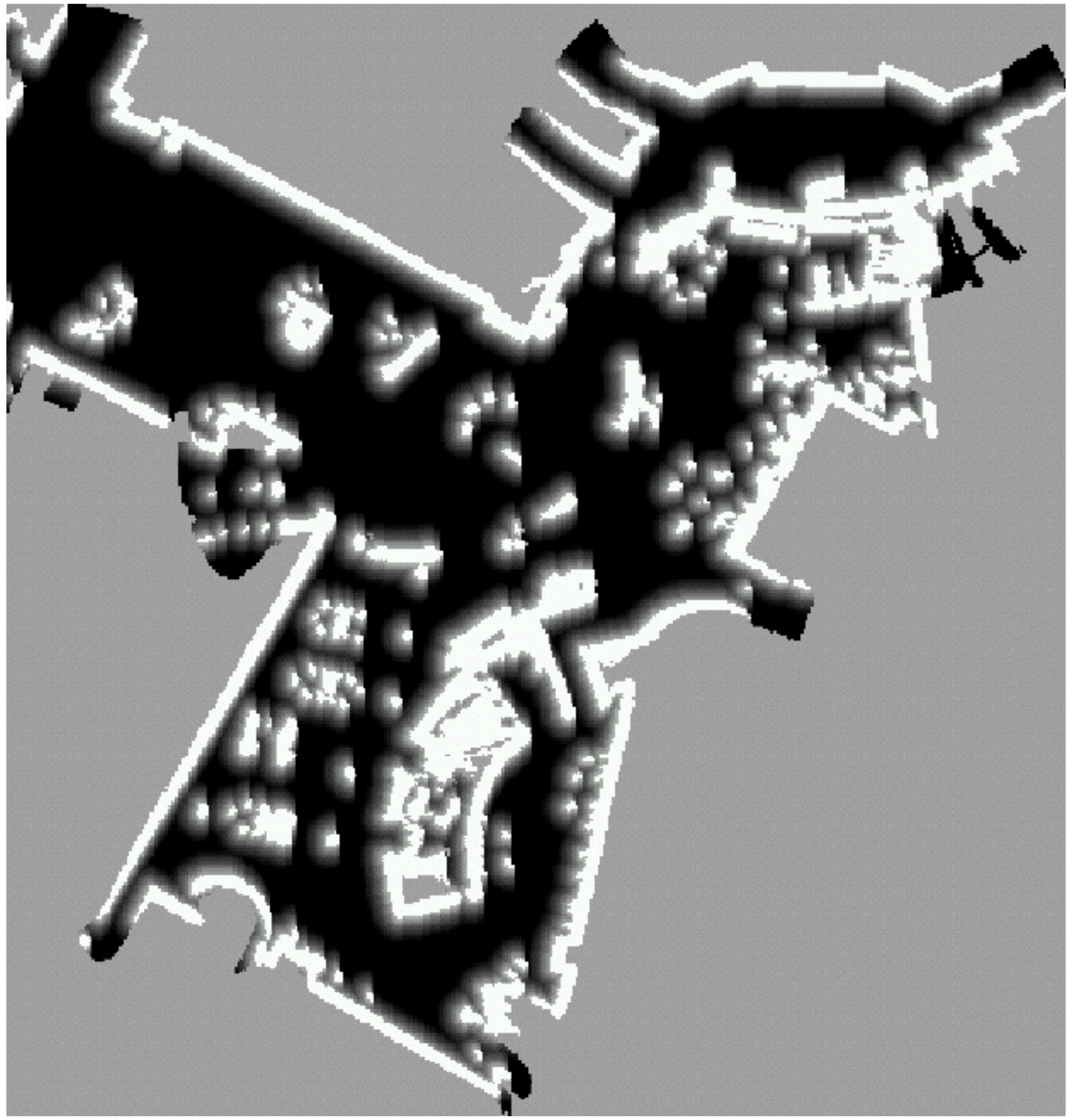
$P(z|x,m)$



San Jose Tech Museum



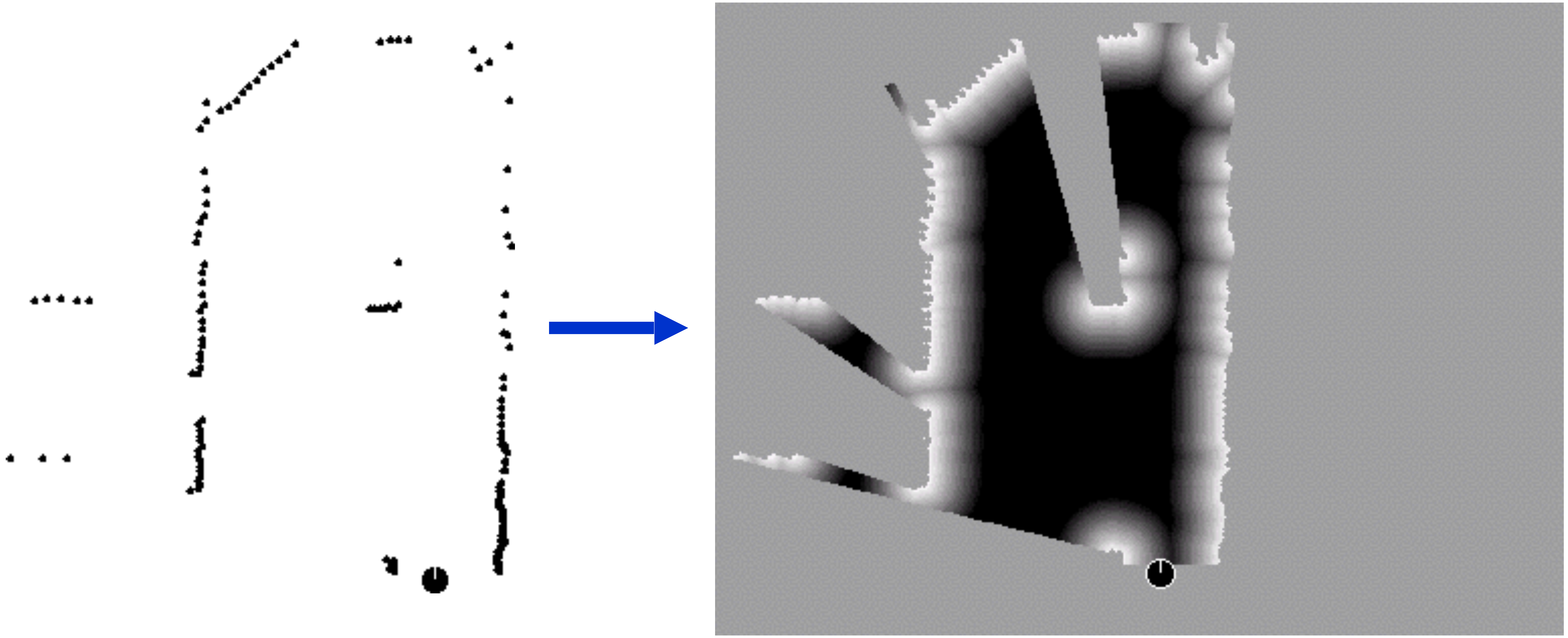
Occupancy grid map



Likelihood field

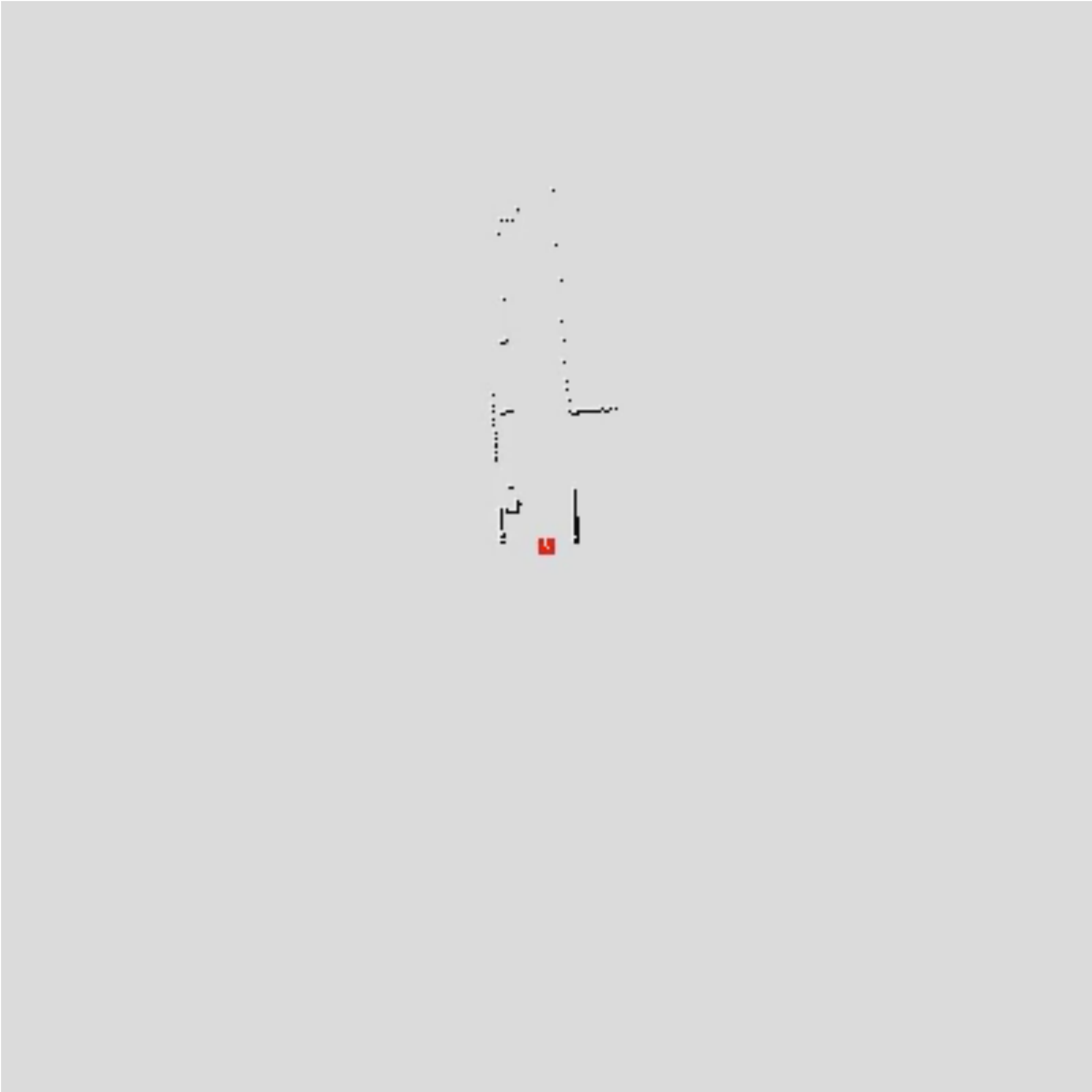
Scan Matching

- Extract likelihood field from scan and use it to match different scan.



Scan Matching

- Extract likelihood field from first scan and use it to match second scan.



~0.01 sec

Properties of Scan-based Model

- Highly efficient, uses 2D tables only.
- Smooth w.r.t. to small changes in robot position.
- Allows gradient descent, scan matching.
- Ignores physical properties of beams.
- Works for sonars?

Additional Models of Proximity Sensors

- Map matching (sonar,laser): generate small, local maps from sensor data and match local maps against global model.

- Scan matching (laser): map is represented by scan endpoints, match scan into this map using ICP, correlation.

- Features (sonar, laser, vision): Extract features such as doors, hallways from sensor data.

Slide borrowed from Dieter Fox

Landmarks

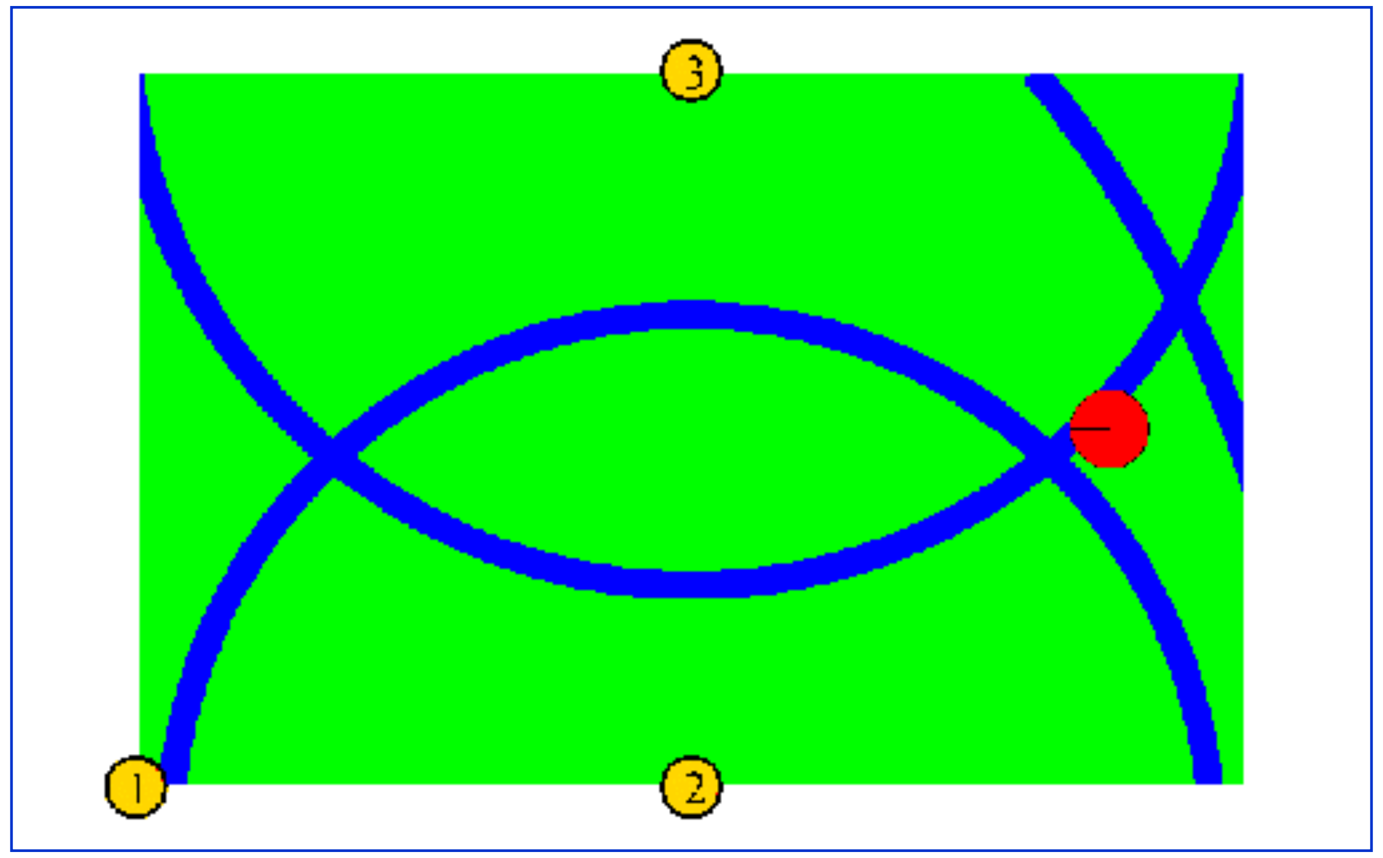
- Active beacons (*e.g*. radio, GPS)
- Passive (*e.g*. visual, retro-reflective)
- Standard approach is triangulation

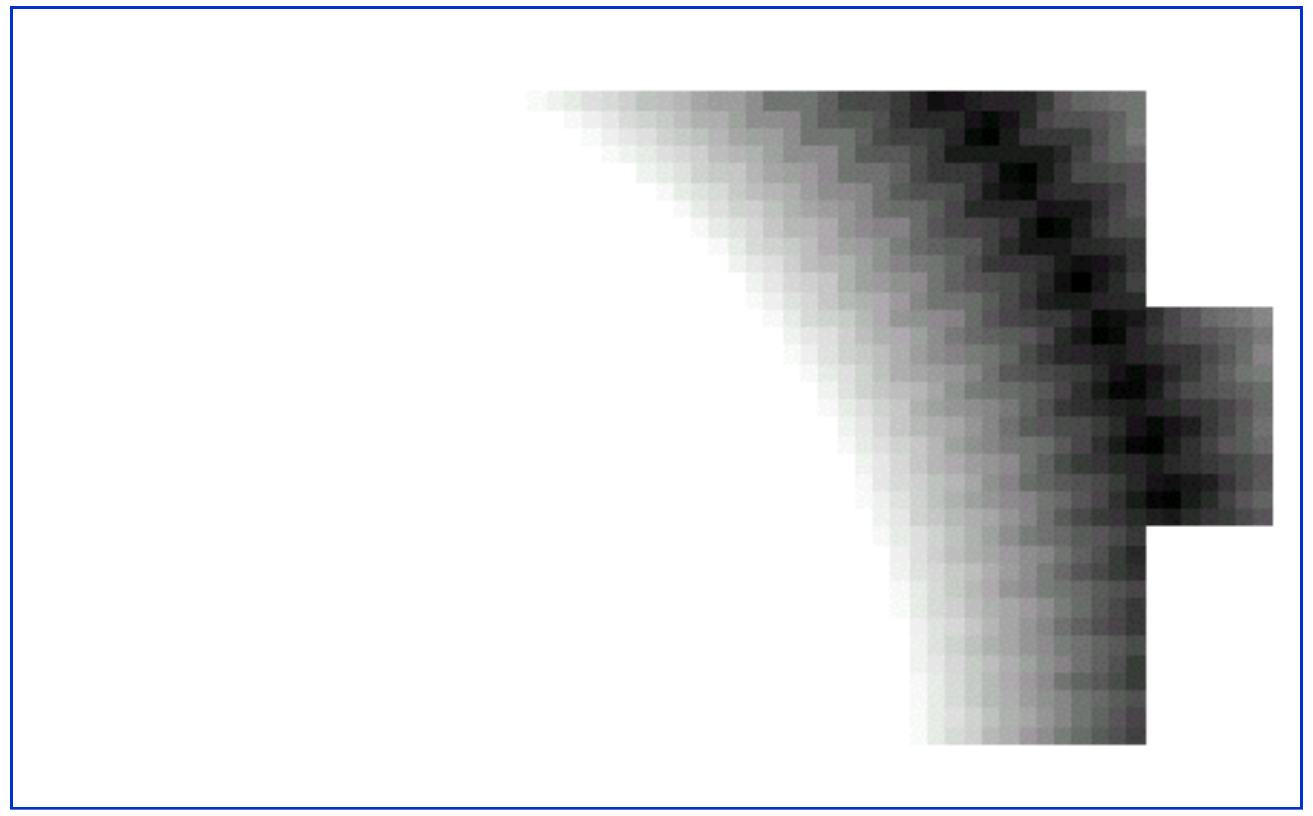
- Sensor provides
  - distance, or
  - bearing, or
  - distance and bearing.

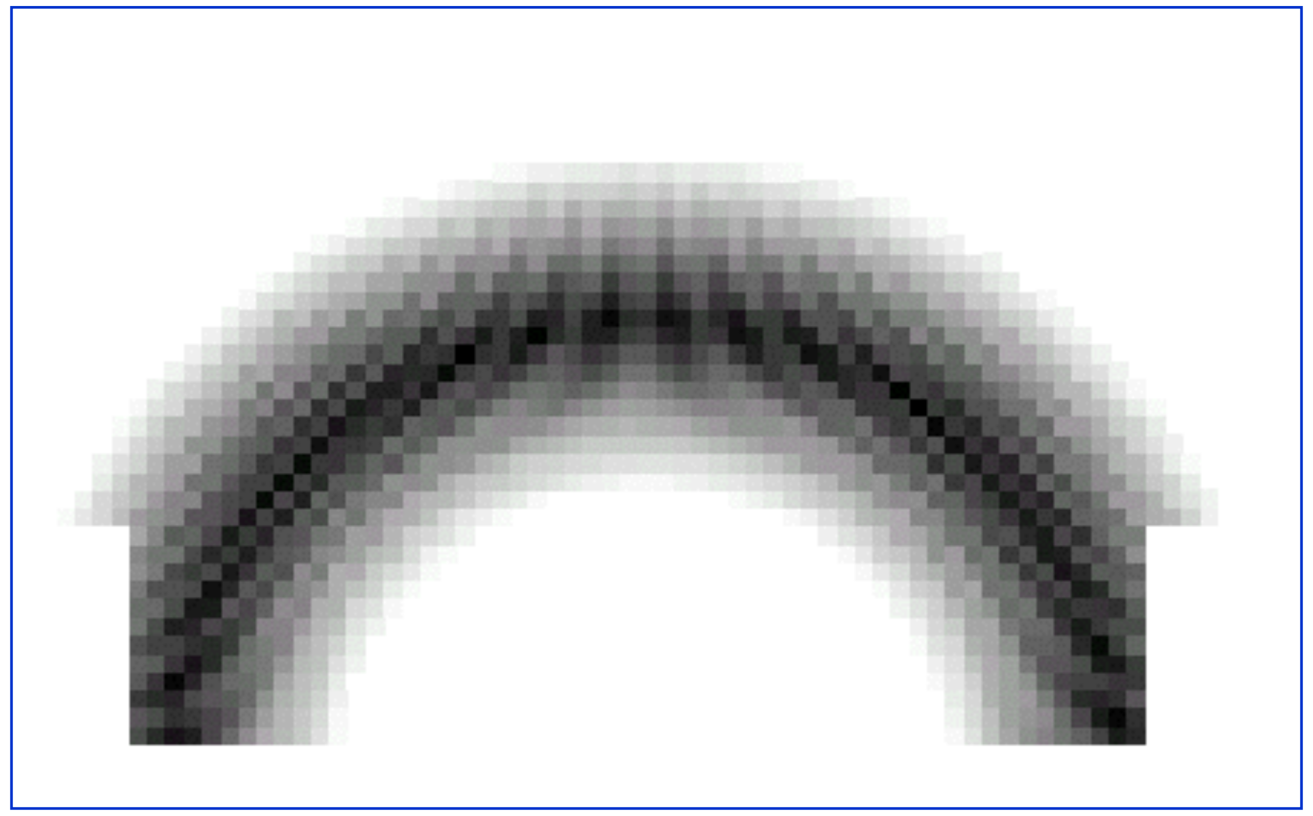
Distance and Bearing

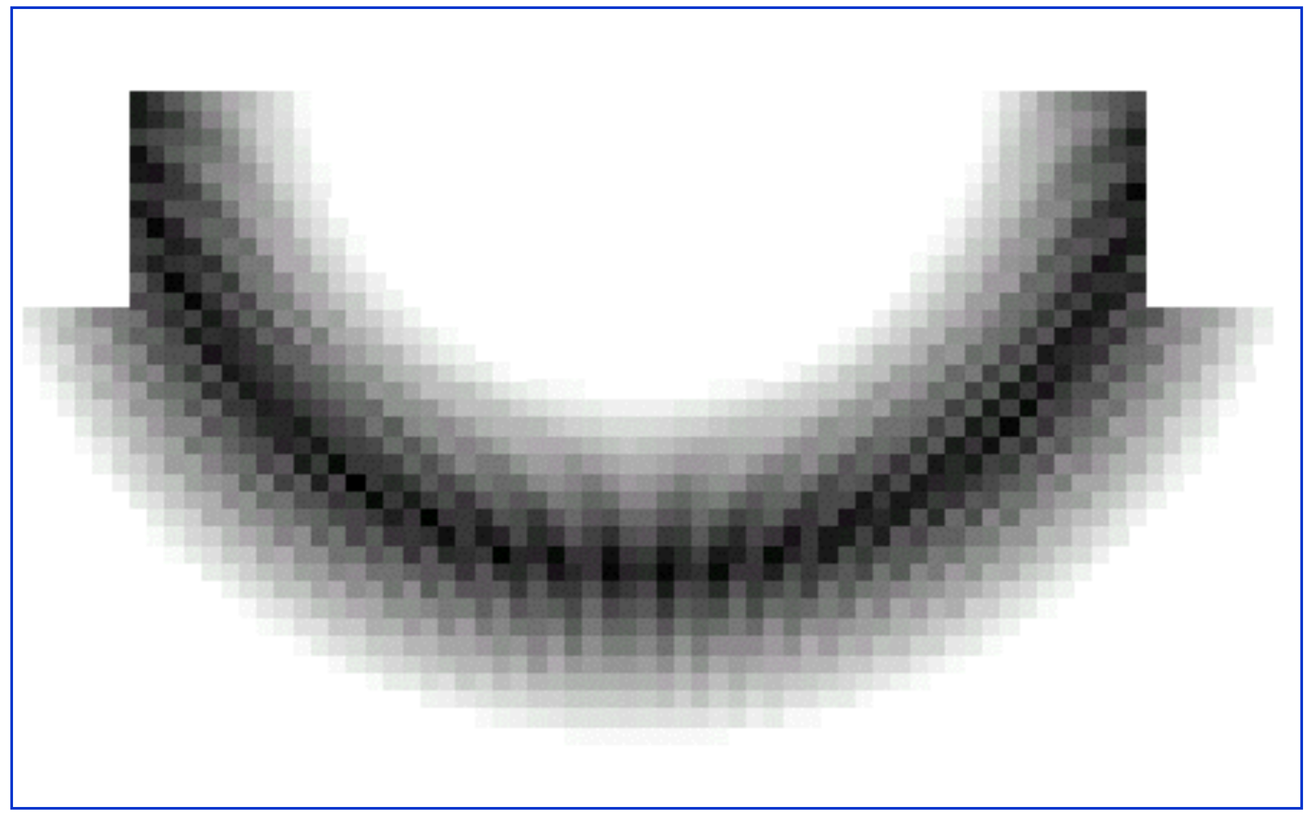
Slide borrowed from Dieter Fox

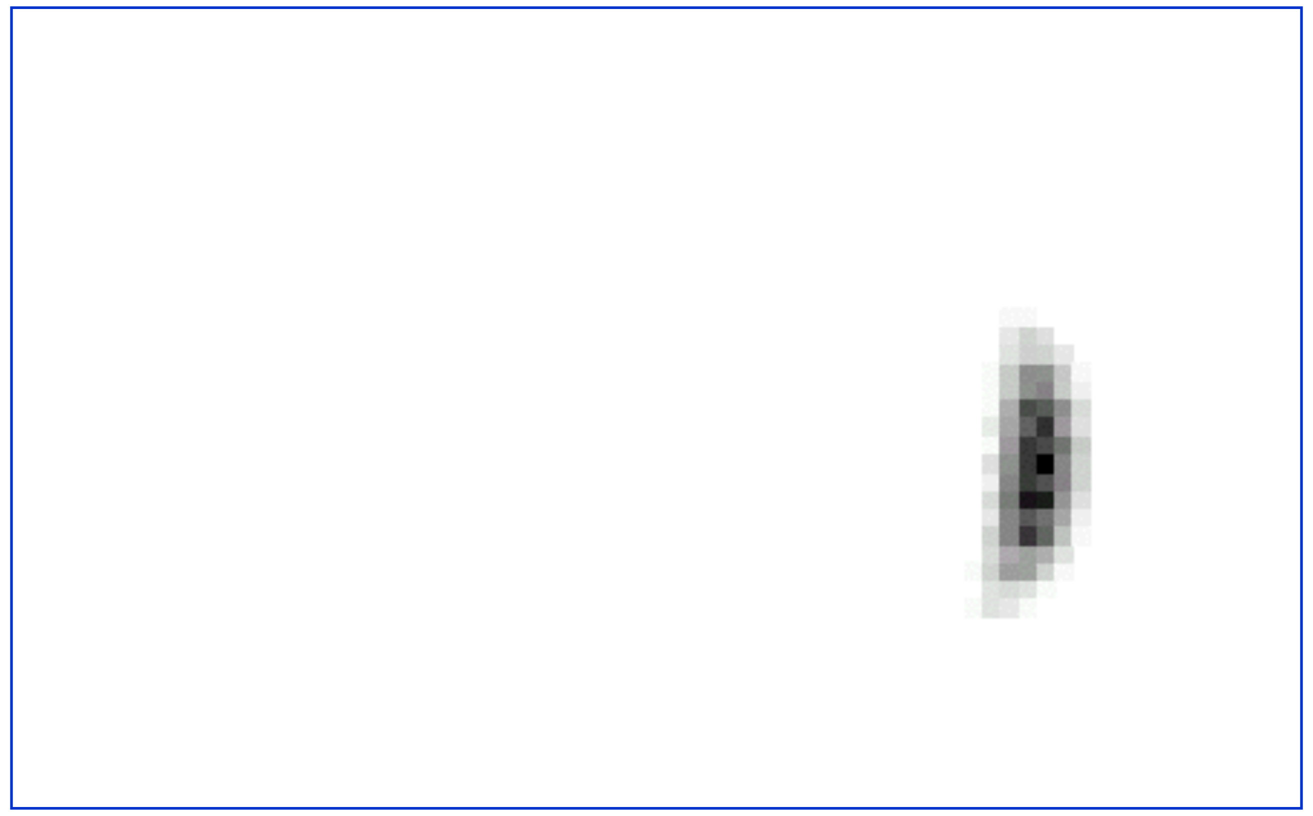
Distributions for P(z|x)











Summary of Parametric Motion and Sensor Models

- Explicitly modeling uncertainty in motion and sensing is key to robustness.
- In many cases, good models can be found by the following approach:
  1. Determine parametric model of noise free motion or measurement.
  2. Analyze sources of noise.
  3. Add adequate noise to parameters (eventually mix densities for noise).
  4. Learn (and verify) parameters by fitting model to data.
  5. Likelihood of measurement is given by "probabilistically comparing" the actual with the expected measurement.
- It is important to be aware of the underlying assumptions!

Mobile Robotics - III - Kalman

Bayes Filter Reminder

- Prediction

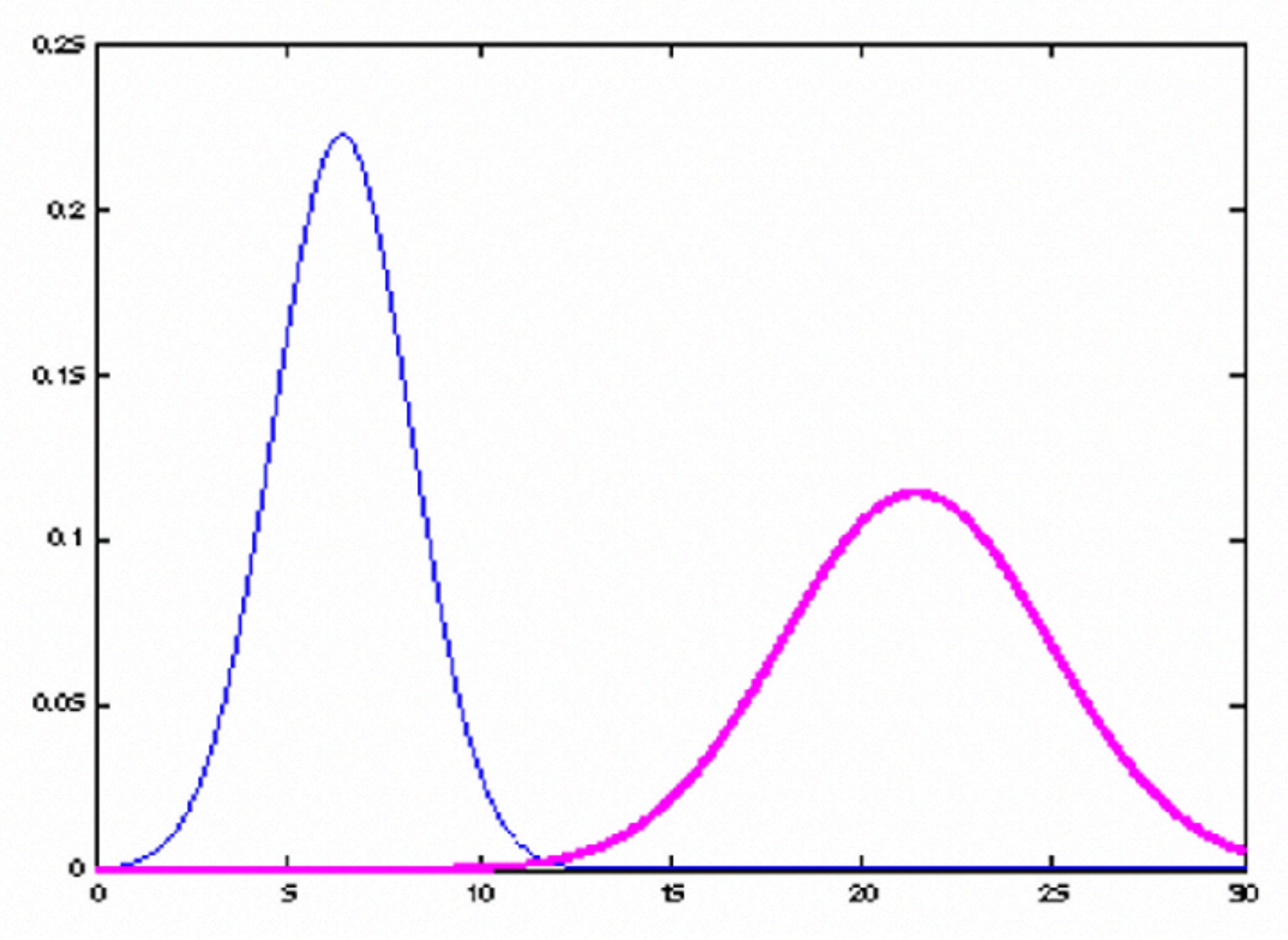
$$\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1})\, bel(x_{t-1})\, dx_{t-1}$$

- Correction

$$bel(x_t) = \eta\, p(z_t \mid x_t)\, \overline{bel}(x_t)$$

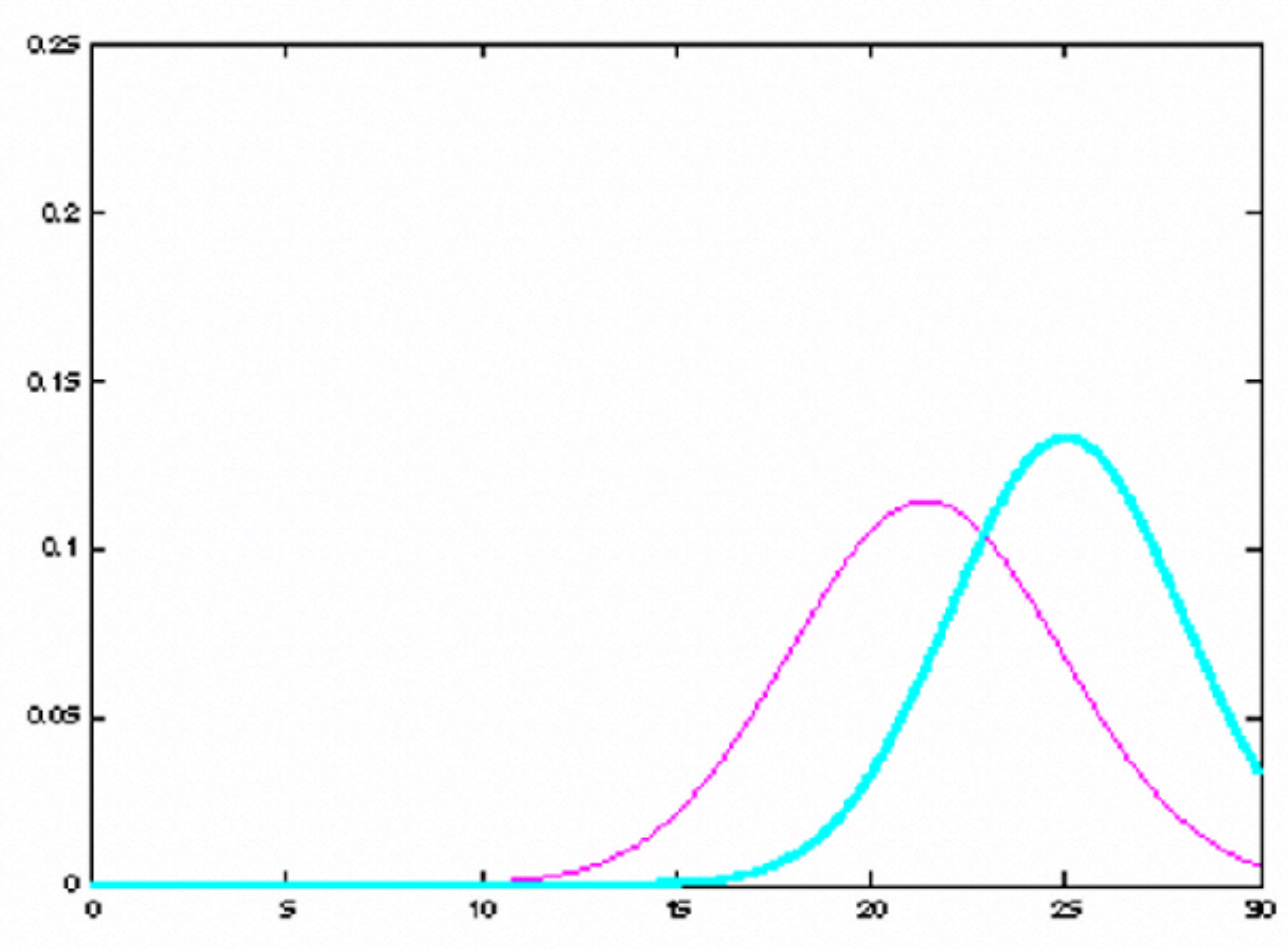
Properties of Gaussians

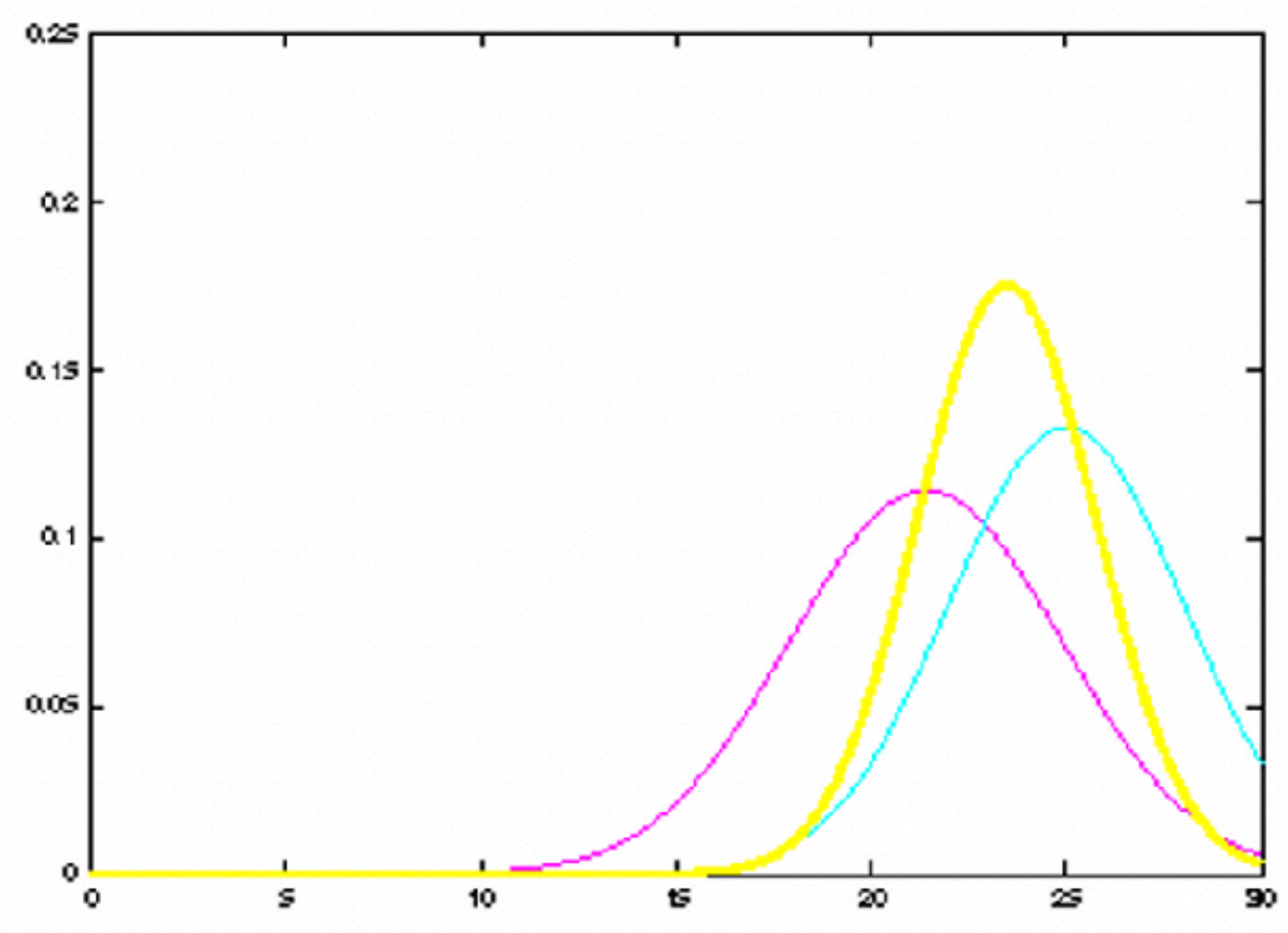
$$\left.\begin{array}{l} X \sim N(\mu, \sigma^2) \\ Y = aX + b \end{array}\right\} \Rightarrow \quad Y \sim N(a\mu + b, a^2\sigma^2)$$



Properties of Gaussians

$$\left.\begin{array}{l} X_1 \sim N(\mu_1, \sigma_1^{\,2}) \\ X_2 \sim N(\mu_2, \sigma_2^{\,2}) \end{array}\right\} \Rightarrow p(X_1) \cdot p(X_2) \sim N\left(\frac{\sigma_2^{\,2}}{\sigma_1^{\,2} + \sigma_2^{\,2}} \mu_1 + \frac{\sigma_1^{\,2}}{\sigma_1^{\,2} + \sigma_2^{\,2}} \mu_2, \quad \frac{1}{\sigma_1^{\,-2} + \sigma_2^{\,-2}} \right)$$





Multivariate Gaussians

$$\left.\begin{array}{l} X \sim N(\mu, \Sigma) \\ Y = AX + B \end{array}\right\} \Rightarrow \quad Y \sim N(A\mu + B, A\Sigma A^T)$$

$$\left.\begin{array}{l} X_1 \sim N(\mu_1, \Sigma_1) \\ X_2 \sim N(\mu_2, \Sigma_2) \end{array}\right\} \Rightarrow p(X_1) \cdot p(X_2) \sim N\left(\frac{\Sigma_2}{\Sigma_1 + \Sigma_2}\mu_1 + \frac{\Sigma_1}{\Sigma_1 + \Sigma_2}\mu_2, \quad \frac{1}{\Sigma_1^{-1} + \Sigma_2^{-1}} \right)$$

- Marginalization and conditioning in Gaussians results in Gaussians
- We stay in the "Gaussian world" as long as we start with Gaussians and perform only linear transformations.

Discrete Kalman Filter

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

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

with a measurement

$$\boxed{z_t = C_t x_t + \delta_t}$$

Components of a Kalman Filter

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

B_t Matrix (nxl) that describes how the control u_t changes the state from t-1 to t.

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

ε_t

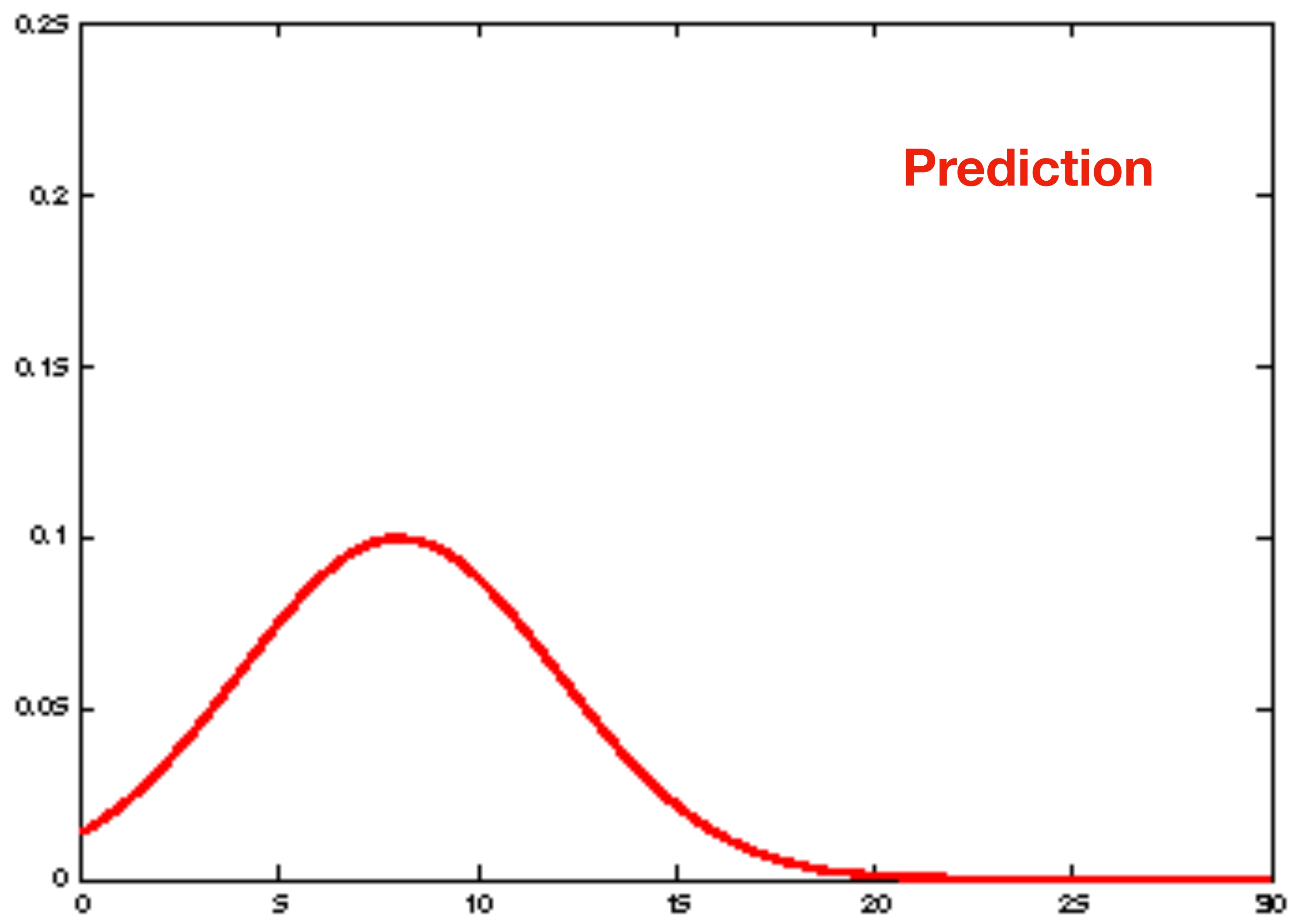
δ_t

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.

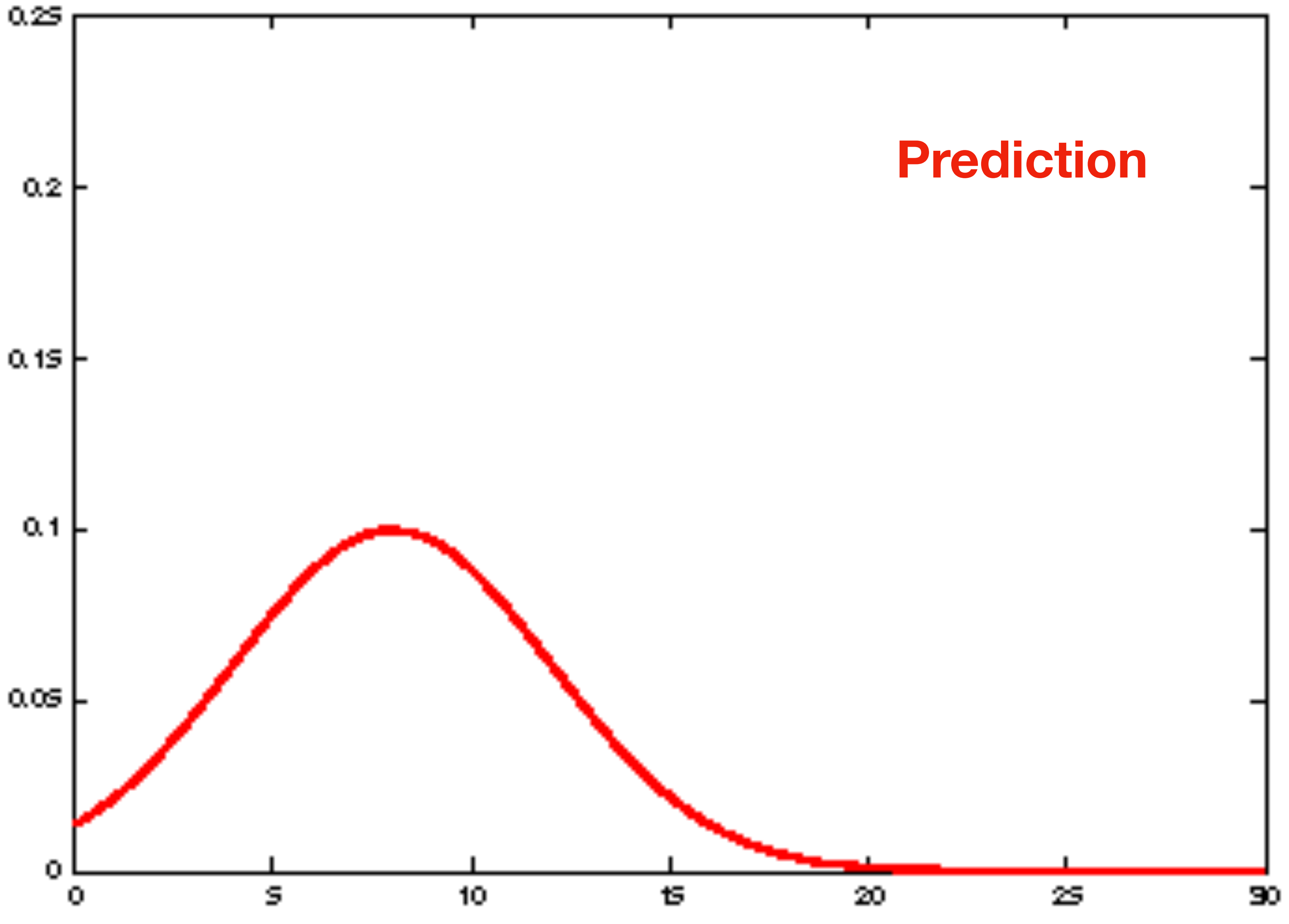
Kalman Filter Updates in 1D

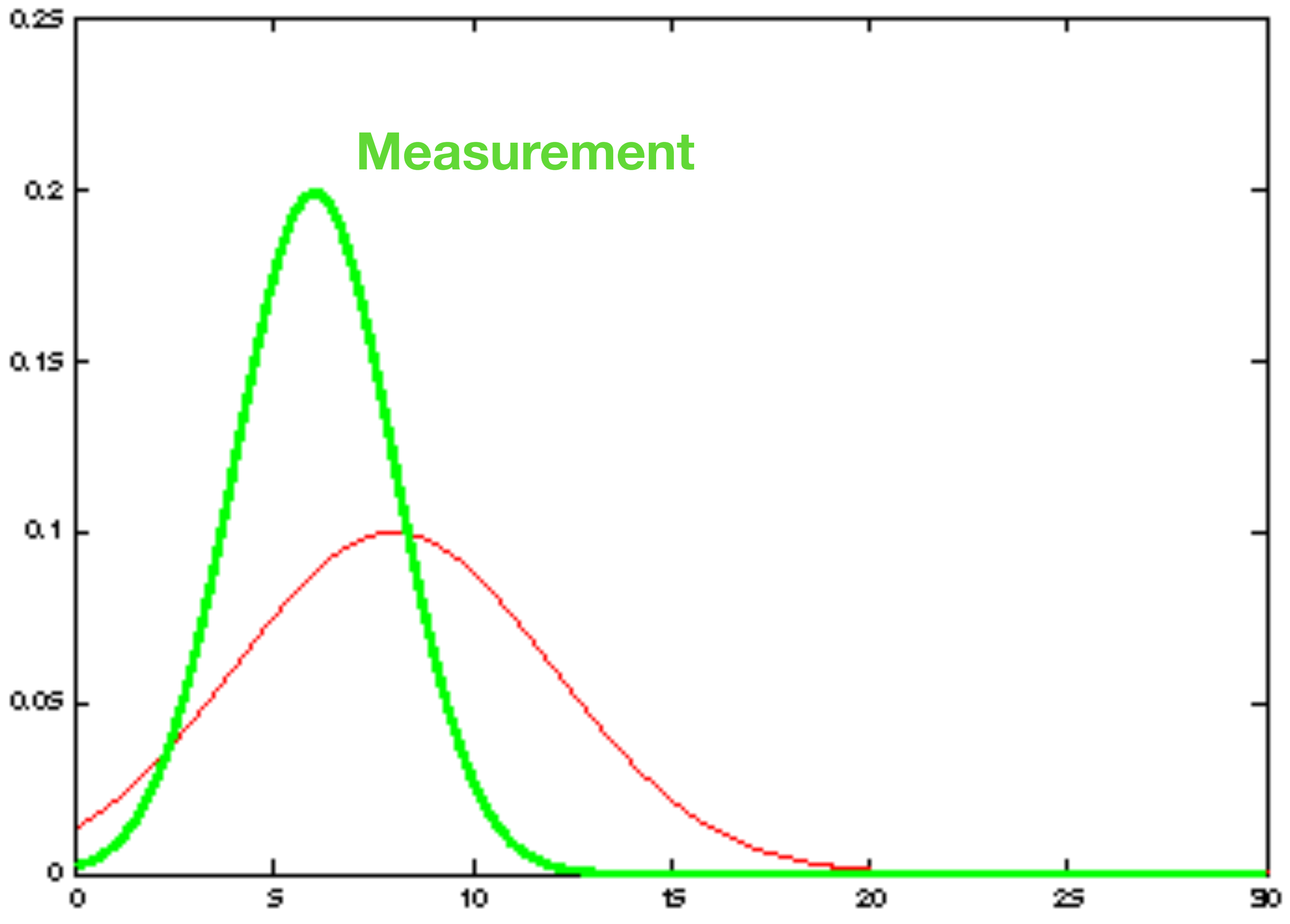
Slide borrowed from Dieter Fox

Kalman Filter Updates in 1D

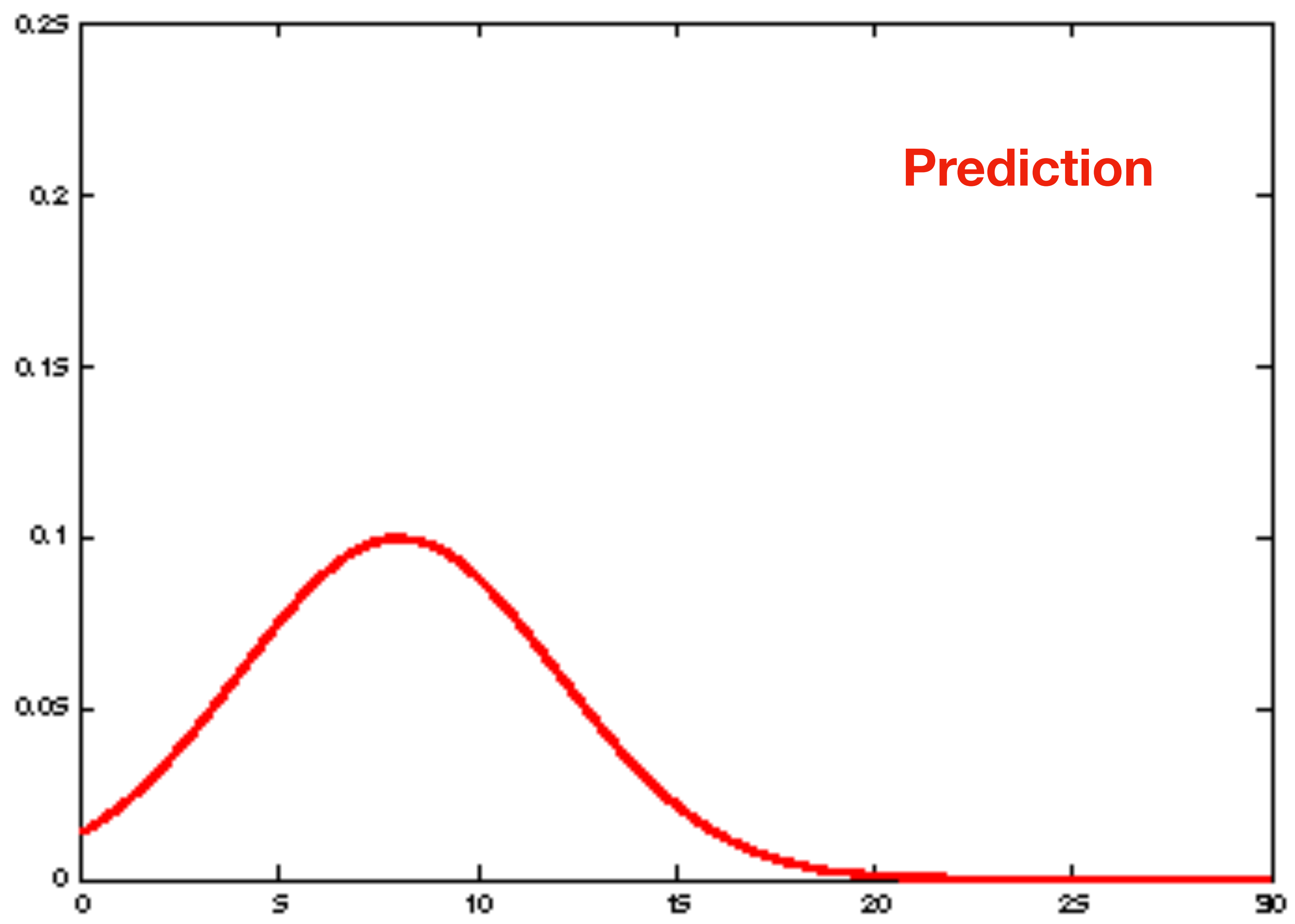


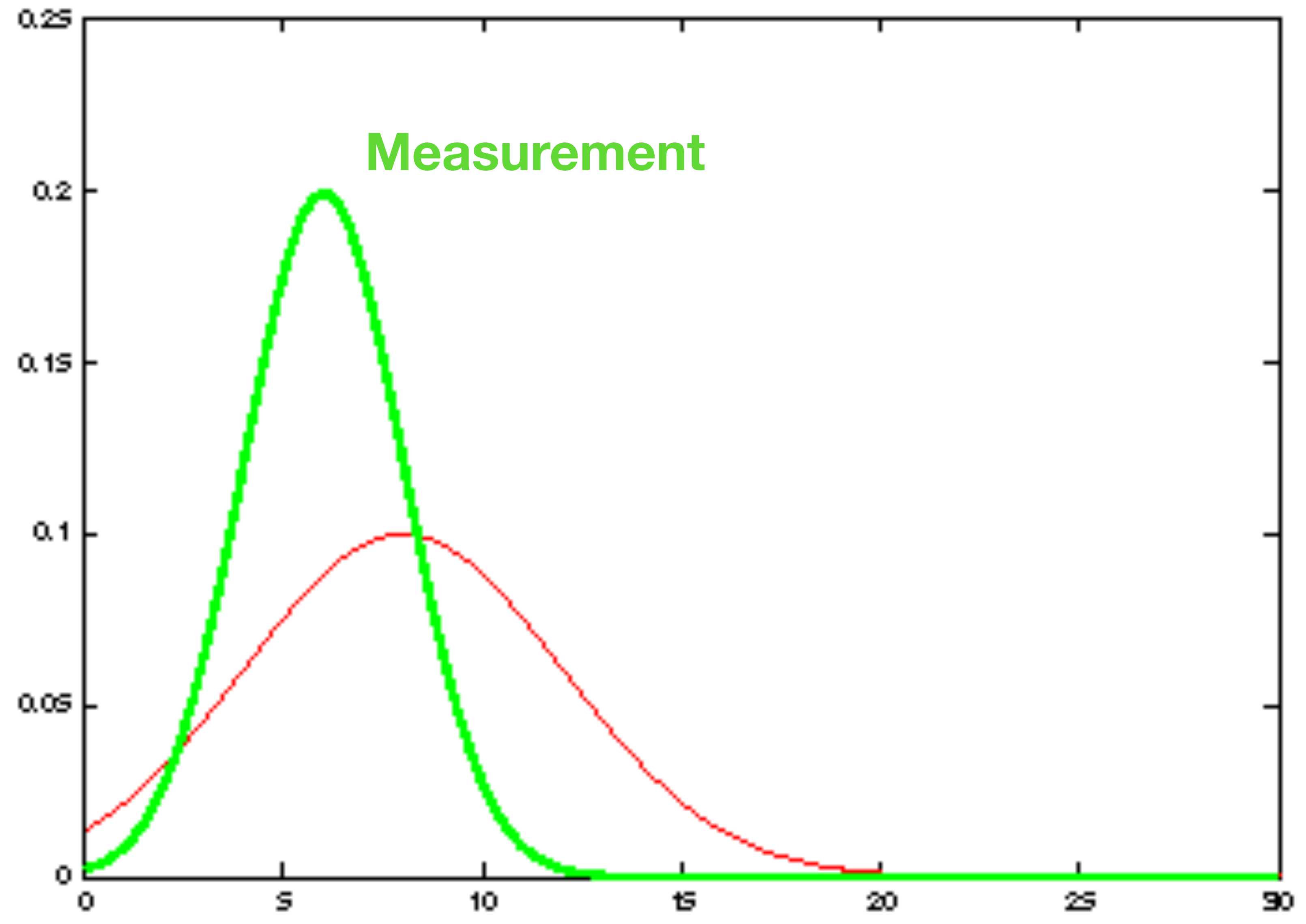
Kalman Filter Updates in 1D

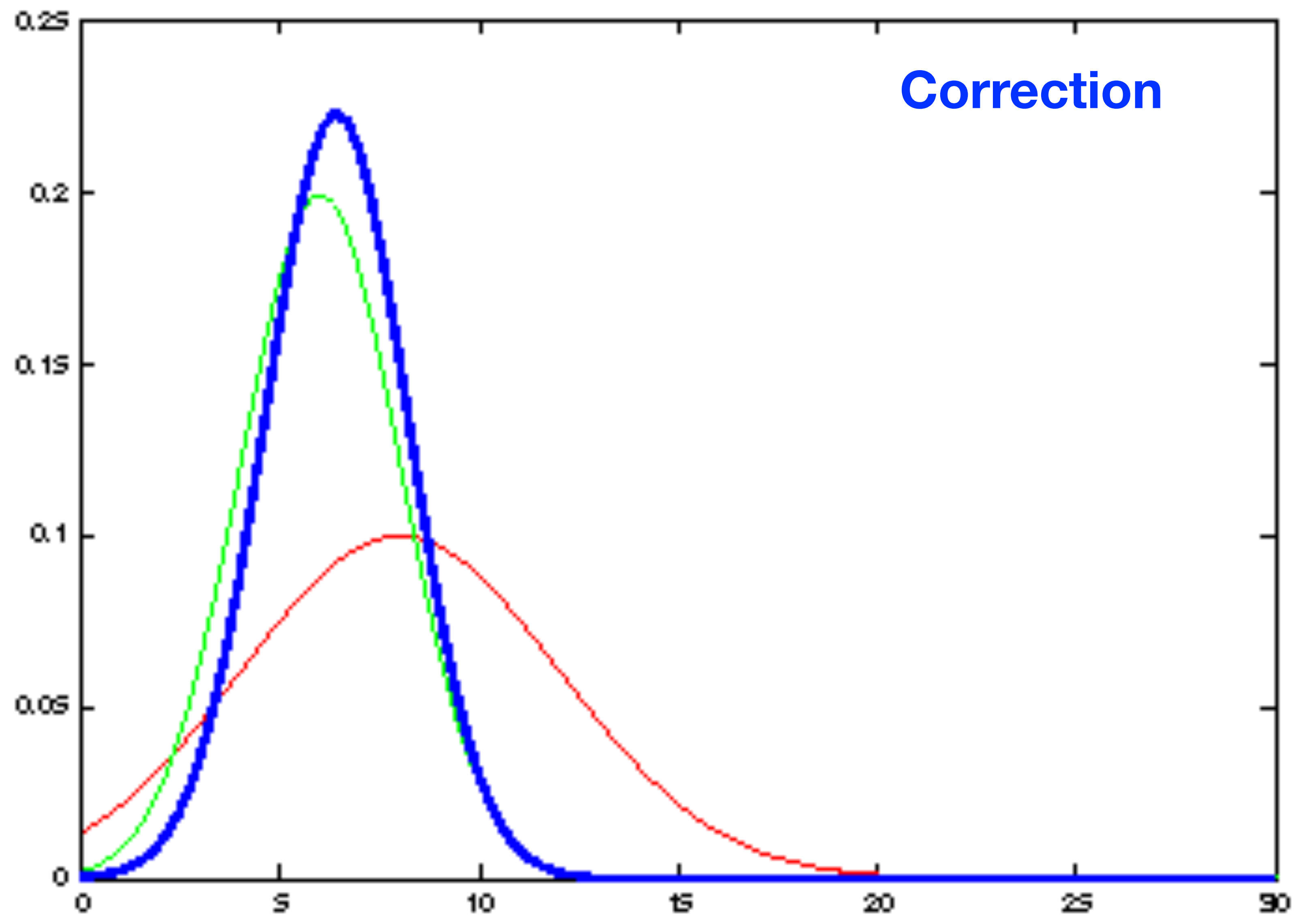




Kalman Filter Updates in 1D





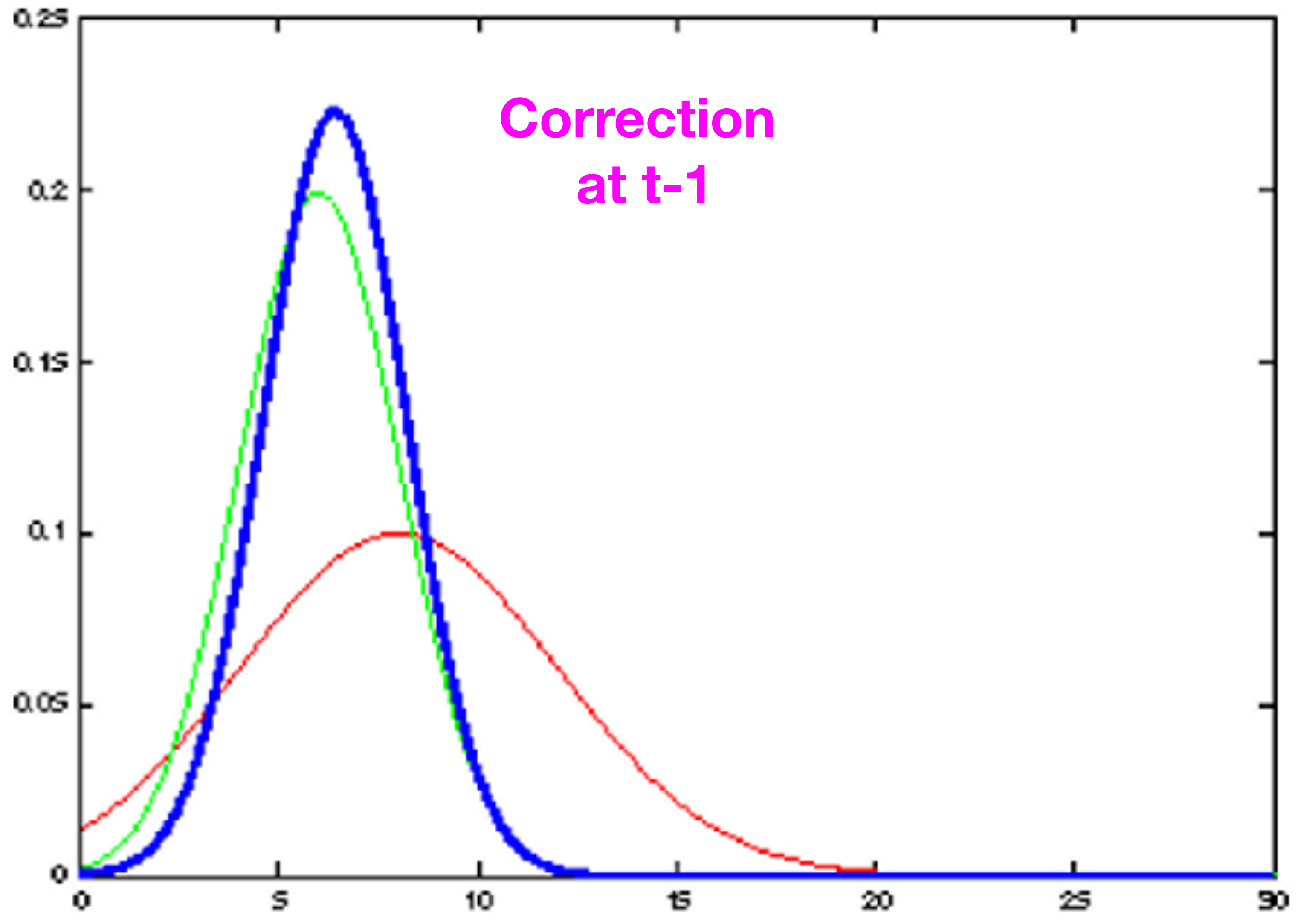


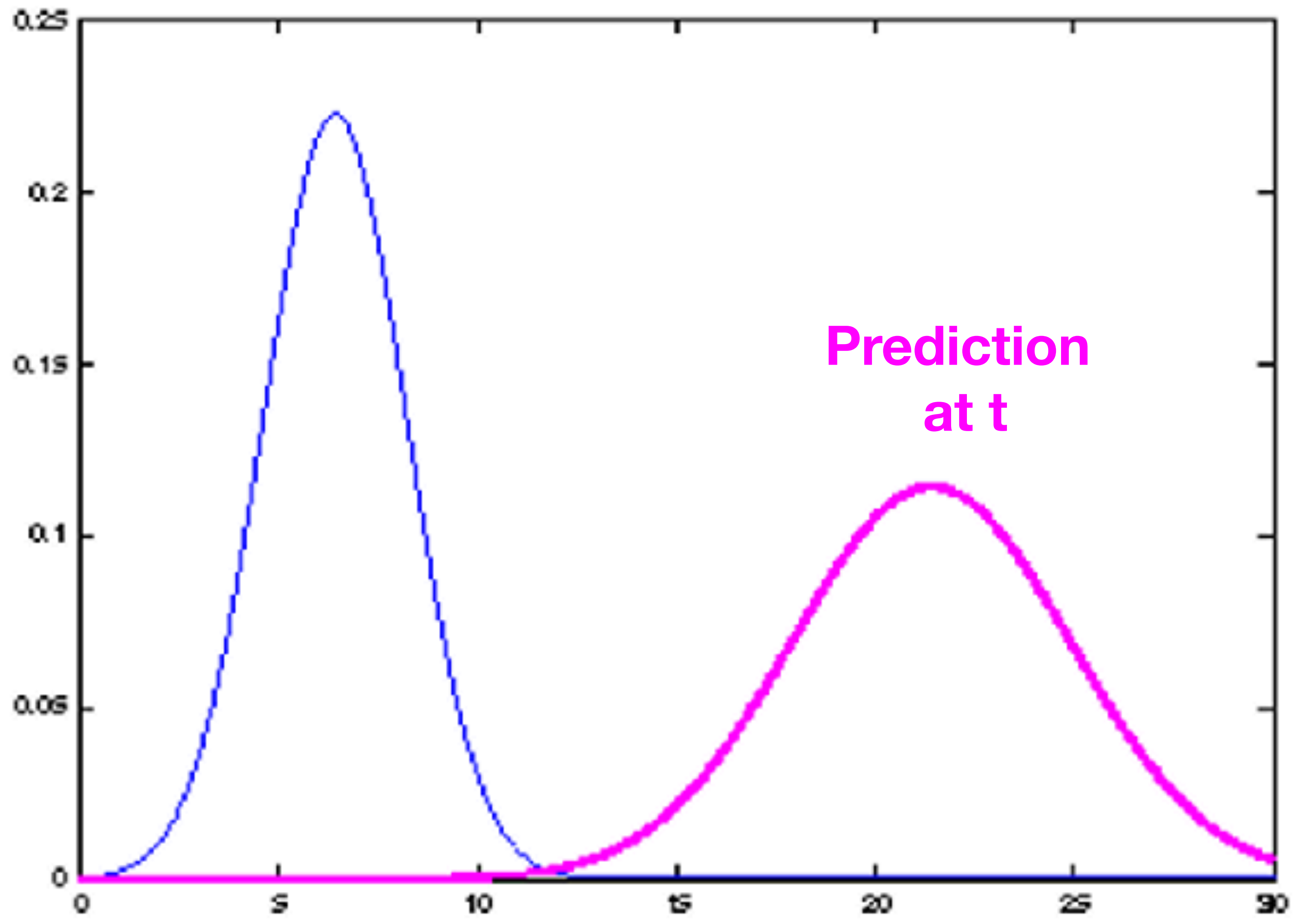
Slide borrowed from Dieter Fox

Kalman Filter Updates in 1D

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = a_t \mu_{t-1} + b_t u_t \\ \bar{\sigma}_t^2 = a_t^2 \sigma_t^2 + \sigma_{act,t}^2 \end{cases}$$

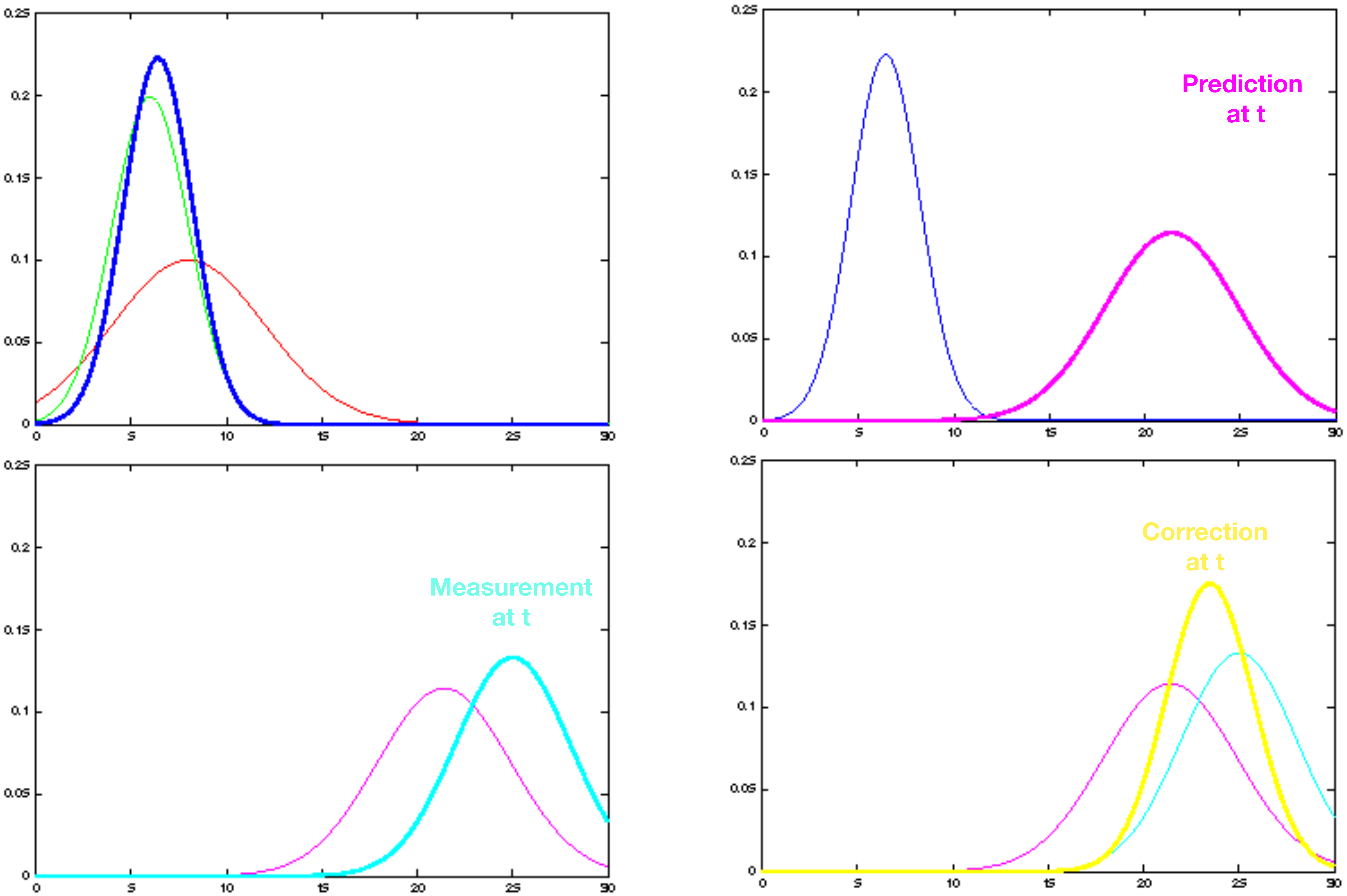
$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \end{cases}$$





Slide borrowed from Dieter Fox

Kalman Filter Updates



Slide borrowed from Dieter Fox

Kalman Filter Algorithm

1. Algorithm **Kalman_filter**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$:

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

5. Correction:
6. $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$
7. $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$
8. $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$

9. Return μ_t, Σ_t

Kalman Filter Summary

- Highly efficient: Polynomial in measurement dimensionality *k* and state dimensionality *n*:

$$O(k^{2.376} + n^2)$$

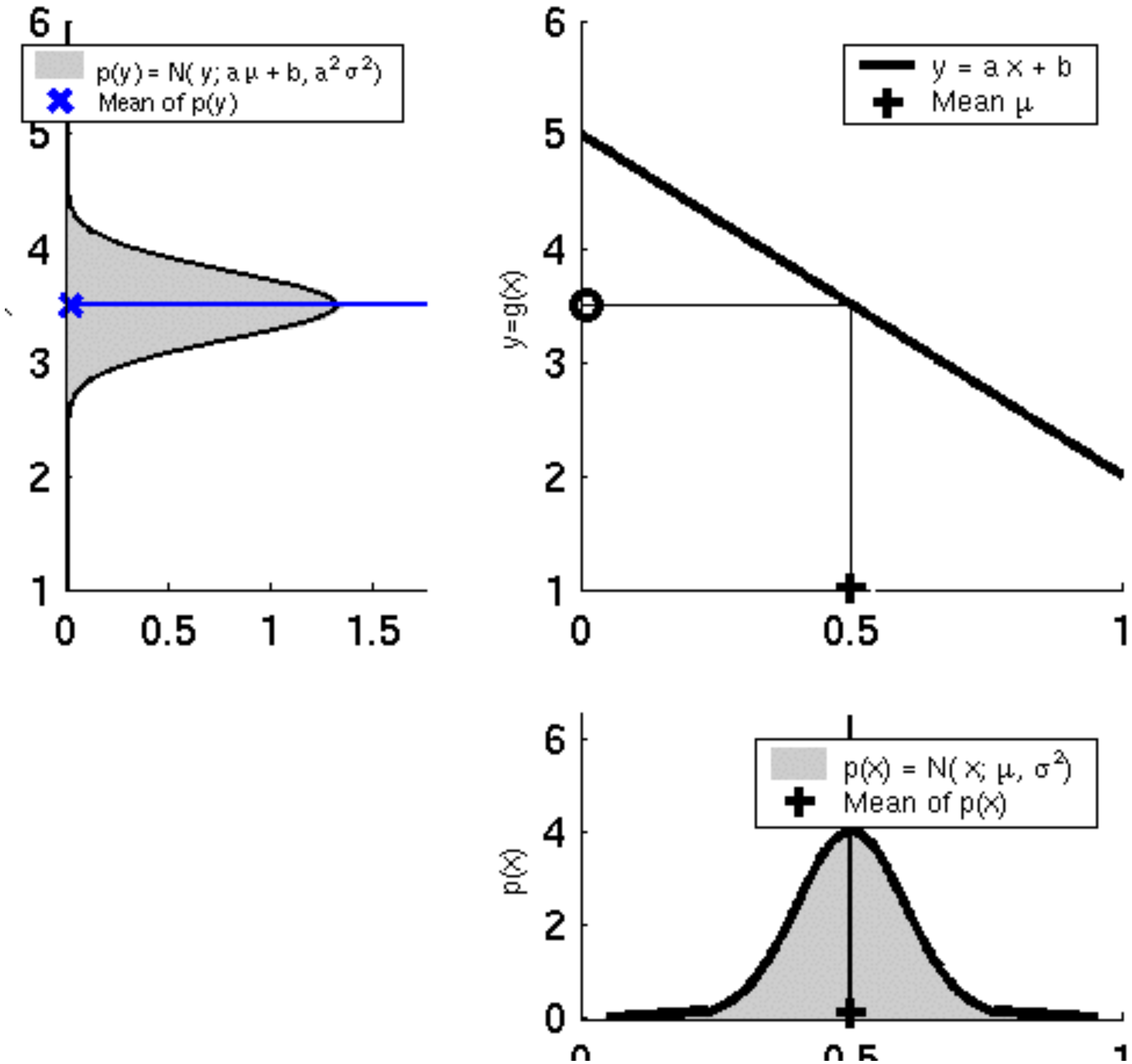
- Optimal for linear Gaussian systems!

- Most robotics systems are nonlinear!

Going non-linear

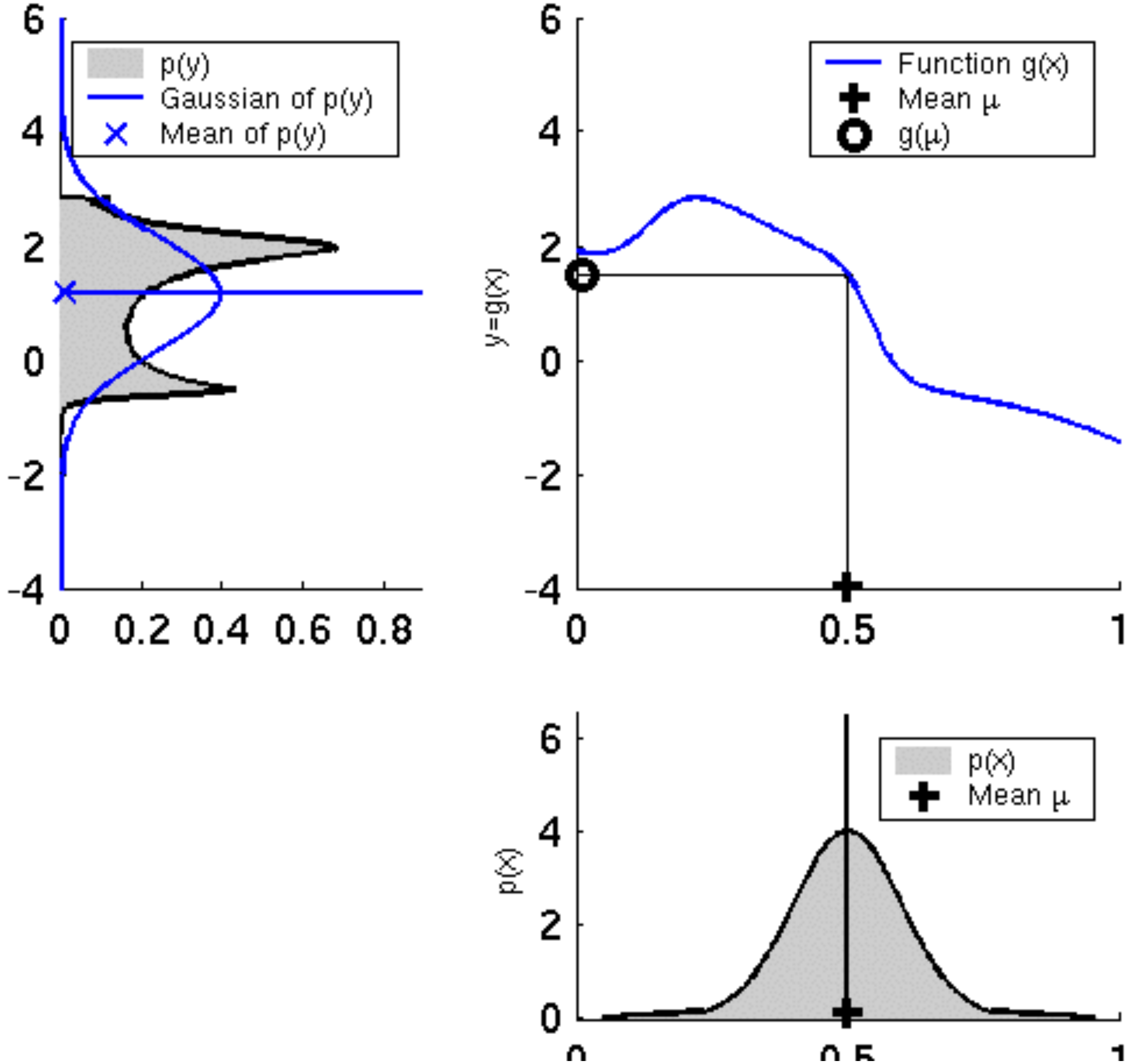
EXTENDED KALMAN FILTER

Linearity Assumption Revisited

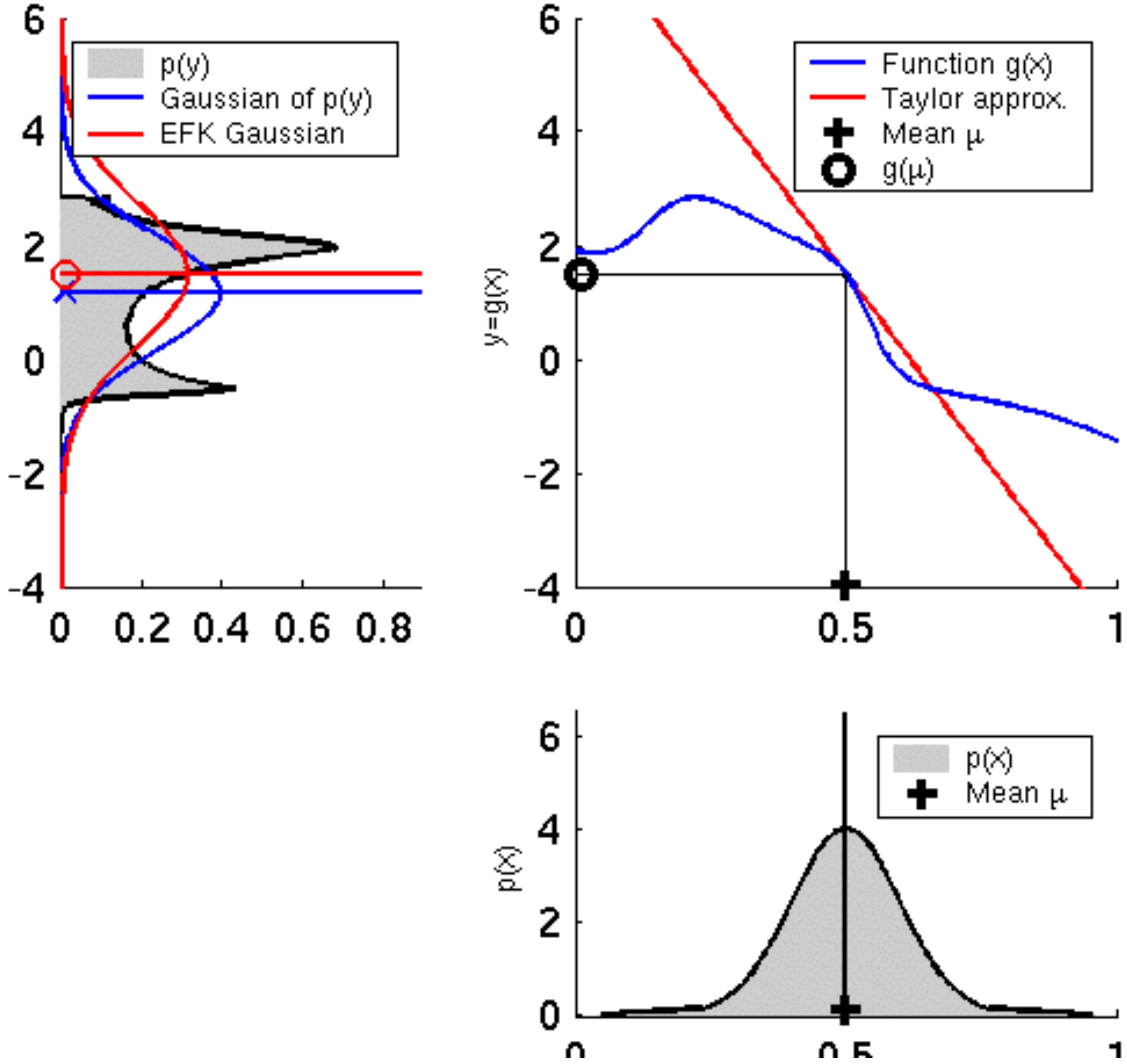


Slide borrowed from Dieter Fox

Non-linear Function

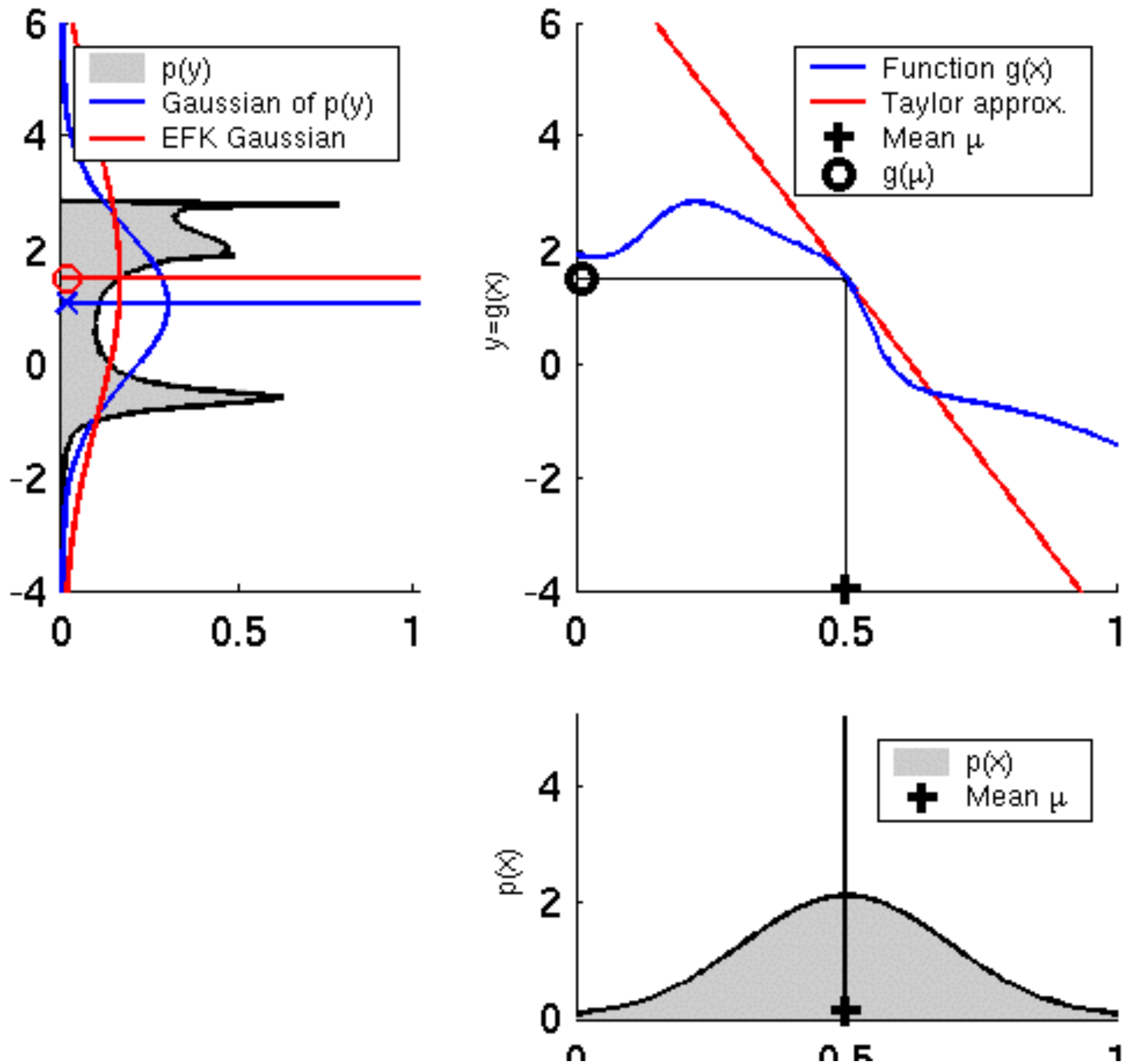


EKF Linearization (1)



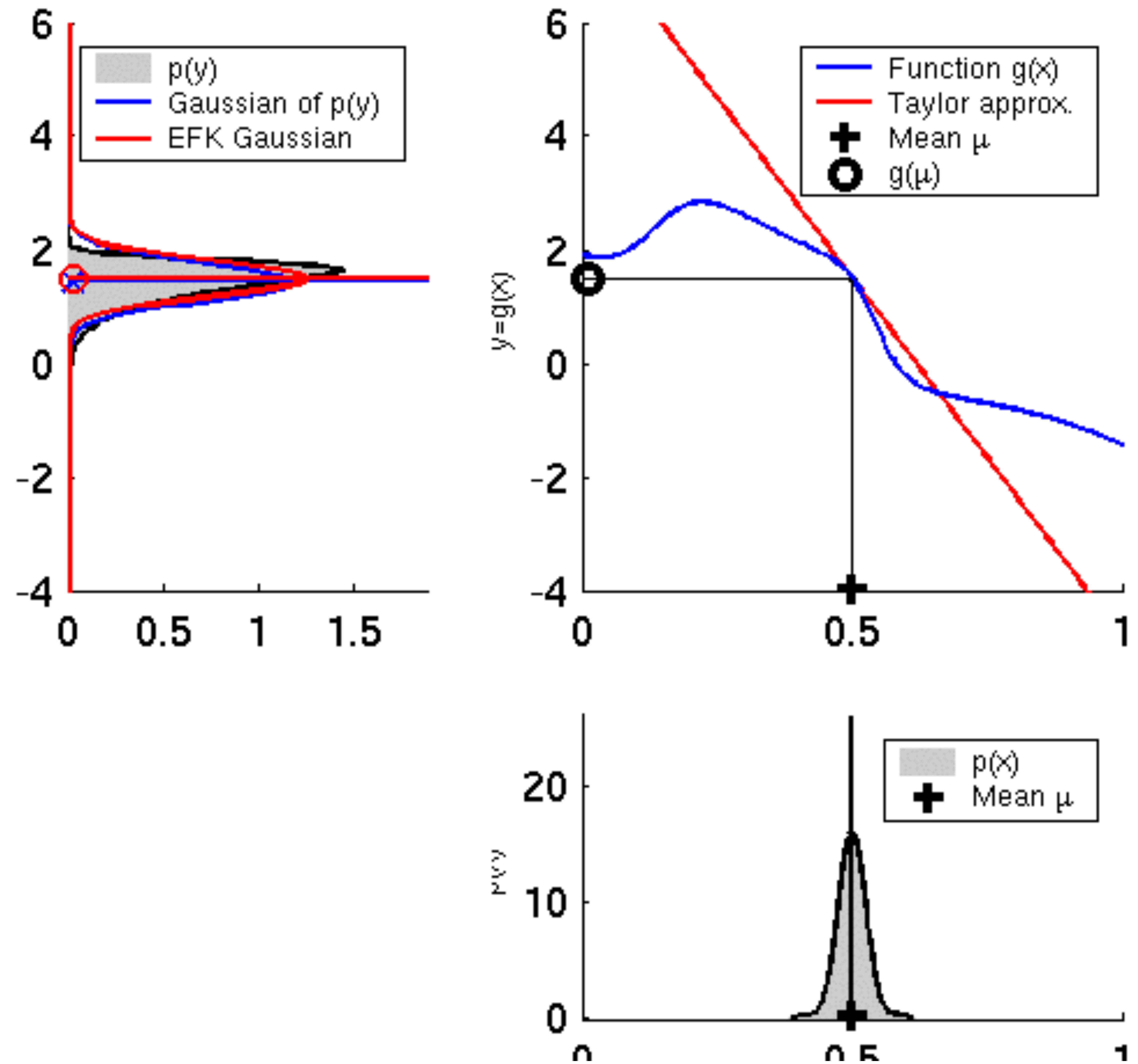
Slide borrowed from Dieter Fox

EKF Linearization (2)



Slide borrowed from Dieter Fox

EKF Linearization (3)



Slide borrowed from Dieter Fox

Linearization

$$x_t = g(u_t, x_{t-1}) + \varepsilon_t \qquad g'(u_t, x_{t-1}) := \frac{\partial g(u_t, x_{t-1})}{\partial x_{t-1}}$$

$$g(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + \underbrace{g'(u_t, \mu_{t-1})}_{=: G_t} (x_{t-1} - \mu_{t-1})$$

$$= g(u_t, \mu_{t-1}) + G_t \, (x_{t-1} - \mu_{t-1})$$

Slide borrowed from Dieter Fox

EKF Algorithm

1. **Extended_Kalman_filter**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$:

2. Prediction:

3. $\bar{\mu}_t = g(u_t, \mu_{t-1})$ $\longleftarrow$ $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$

4. $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$ $\longleftarrow$ $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$

5. Correction:

6. $K_t = \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + Q_t)^{-1}$ $\longleftarrow$ $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$

7. $\mu_t = \bar{\mu}_t + K_t (z_t - h(\bar{\mu}_t))$ $\longleftarrow$ $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$

8. $\Sigma_t = (I - K_t H_t) \bar{\Sigma}_t$ $\longleftarrow$ $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$

9. Return μ_t, Σ_t

$$H_t = \frac{\partial h(\bar{\mu}_t)}{\partial x_t} \qquad G_t = \frac{\partial g(u_t, \mu_{t-1})}{\partial x_{t-1}}$$

Localization

> "Using sensory information to locate the robot in its environment is the most fundamental problem to providing a mobile robot with autonomous capabilities." [Cox '91]

- **Given**
  - Map of the environment.
  - Sequence of sensor measurements.
- **Wanted**
  - Estimate of the robot's position.
- **Problem classes**
  - Position tracking
  - Global localization
  - Kidnapped robot problem (recovery)

Next Lecture
Localization & Particle Filter