732A96/TDDE15 Advanced Machine Learning

State Space Models

Jose M. Peña
IDA, Linköping University, Sweden

Lecture 10: Linear-Gaussian State Space Models and the Kalman Filter

Contents

- Linear Gaussian State Space Models
- Robot Localization
- Bayes Filter
- Kalman Filter

Literature

- Main source
  - Thrun, S. et al. *Probabilistic Robotics*. MIT Press, 2005. Chapters 2, 3.1 and 3.2.
- Additional source
  - Bishop, C. M. *Pattern Recognition and Machine Learning*. Springer, 2006. Chapter 13.3.

Linear Gaussian State Space Models

- SSMs = HMMs where the latent variables are continuous.

Linear Gaussian State Space Models

- SSMs = HMMs where the latent variables are continuous.
- Moreover, we assume that the transition, emission and initial distributions are Gaussian. That is,

$$
\begin{aligned}
p(z_t|z_{t-1}) &= \mathcal{N}(z_t|Az_{t-1}, \Gamma) \\
p(x_t|z_t) &= \mathcal{N}(x_t|Cz_t, \Sigma) \\
p(z_0) &= \mathcal{N}(z_0|\mu_0, V_0)
\end{aligned}
$$

Linear Gaussian State Space Models

- SSMs = HMMs where the latent variables are continuous.
- Moreover, we assume that the transition, emission and initial distributions are Gaussian. That is,

$$
\begin{aligned}
p(z_t|z_{t-1}) &= \mathcal{N}(z_t|Az_{t-1}, \Gamma) \\
p(x_t|z_t) &= \mathcal{N}(x_t|Cz_t, \Sigma) \\
p(z_0) &= \mathcal{N}(z_0|\mu_0, V_0)
\end{aligned}
$$

or equivalently

$$
\begin{aligned}
z_t &= Az_{t-1} + w_t && \text{// Linear model} \\
x_t &= Cz_t + v_t \\
z_0 &= \mu_0 + u_0
\end{aligned}
$$

where

$$
\begin{aligned}
w_t &\sim \mathcal{N}(w_t|0, \Gamma) && \text{// Gaussian noise} \\
v_t &\sim \mathcal{N}(v_t|0, \Sigma) \\
u_0 &\sim \mathcal{N}(u_0|0, V_0)
\end{aligned}
$$

because recall that $E[Ax+B] = AE[x]+B$ and $cov[Ax+B] = Acov[x]A^T$.

Linear Gaussian State Space Models

- Recall that if

$$p(x) = \mathcal{N}(x|\mu, \Lambda^{-1})$$
$$p(y|x) = \mathcal{N}(y|Ax + B, L^{-1})$$

then

$$p(x, y) = \mathcal{N}(x, y|A\mu + B, R^{-1})$$

where

$$R^{-1} = \begin{pmatrix} \Lambda^{-1} & \Lambda^{-1}A^T \\ A\Lambda^{-1} & L^{-1} + A\Lambda^{-1}A^T \end{pmatrix}$$

Linear Gaussian State Space Models

- Recall that if

$$p(x) = \mathcal{N}(x|\mu, \Lambda^{-1})$$
$$p(y|x) = \mathcal{N}(y|Ax + B, L^{-1})$$

then

$$p(x, y) = \mathcal{N}(x, y|A\mu + B, R^{-1})$$

where

$$R^{-1} = \begin{pmatrix} \Lambda^{-1} & \Lambda^{-1}A^T \\ A\Lambda^{-1} & L^{-1} + A\Lambda^{-1}A^T \end{pmatrix}$$

- Recall also that if $p(x) = \mathcal{N}(x|\mu, \Sigma)$ and $\Lambda = \Sigma^{-1}$ and

$$x = (x_a, x_b)^T \qquad \mu = (\mu_a, \mu_b)^T$$
$$\Sigma = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix} \qquad \Lambda = \begin{pmatrix} \Lambda_{aa} & \Lambda_{ab} \\ \Lambda_{ba} & \Lambda_{bb} \end{pmatrix}$$

then

$$p(x_a) = \mathcal{N}(x_a|\mu_a, \Sigma_{aa})$$
$$p(x_a|x_b) = \mathcal{N}(x_a|\mu_{a|b}, \Lambda_{aa}^{-1}) \text{ where } \mu_{a|b} = \mu_a - \Lambda_{aa}^{-1}\Lambda_{ab}(x_b - \mu_b)$$

Linear Gaussian State Space Models

- Note that the filtered and smoothed distributions, i.e. $p(z^t|x^{0:t})$ and $p(z^t|x^{0:T})$, are Gaussian and we know how to compute them from the transition, emission and initial distributions:
  - Build the joint distribution $p(z^{0:t}, x^{0:t})$ and $p(z^{0:t}, x^{0:T})$ and, then, marginalize and condition.
  - Then, we know how to reason with linear Gaussian SSMs.

Linear Gaussian State Space Models

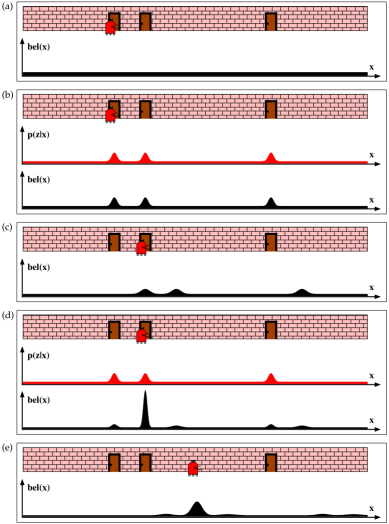
- Note that the filtered and smoothed distributions, i.e. $p(z^t|x^{0:t})$ and $p(z^t|x^{0:T})$, are Gaussian and we know how to compute them from the transition, emission and initial distributions:
  - Build the joint distribution $p(z^{0:t}, x^{0:t})$ and $p(z^{0:t}, x^{0:T})$ and, then, marginalize and condition.
  - Then, we know how to reason with linear Gaussian SSMs.
- We also know how to compute $\alpha(z^t)$ and $\beta(z^t)$:
  - Simply replace summations with integrals in the equations for HMMs.
  - Then, we know how to learn linear Gaussian SSMs: EM + FB algorithms.

Linear Gaussian State Space Models

- Note that the filtered and smoothed distributions, i.e. $p(z^t|x^{0:t})$ and $p(z^t|x^{0:T})$, are Gaussian and we know how to compute them from the transition, emission and initial distributions:
  - Build the joint distribution $p(z^{0:t}, x^{0:t})$ and $p(z^{0:t}, x^{0:T})$ and, then, marginalize and condition.
  - Then, we know how to reason with linear Gaussian SSMs.
- We also know how to compute $\alpha(z^t)$ and $\beta(z^t)$:
  - Simply replace summations with integrals in the equations for HMMs.
  - Then, we know how to learn linear Gaussian SSMs: EM + FB algorithms.
- Note that the Viterbi algorithm is not needed: The most probable path consists of the most probable values of the smoothed distributions, because no transition has zero probability.

Robot Localization

- The robot may need to know itself its location, e.g. to plan ahead.



x = latent variable
z = observed variable
$bel(x^t) = p(x^t | z^{0:t})$ = filtering
Confussing notation, because Thrun et al. invert Bishop's notation. We follow the former since their book is the primary source for this lecture.

Robot Localization

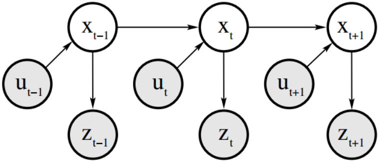


Figure 2.2 The dynamic Bayes network that characterizes the evolution of controls, states, and measurements.

- x_t = state, e.g. robot's position.
- z_t = measurement, e.g. robot's sensor reading.
- u_t = control, e.g. robot's action.
- State transition probability = $p(x'|u, x)$.
- Measurement probability = $p(z|x)$.
- Note the Markovian and stationary assumptions.
- Belief $bel(x_t) = p(x_t|z_{1:t}, u_{1:t})$.
- Prior belief $\overline{bel}(x_t) = p(x_t|z_{1:t-1}, u_{1:t})$.
- Prediction = computing $\overline{bel}(x_t)$ from $bel(x_{t-1})$.
- Correction = computing $bel(x_t)$ from $\overline{bel}(x_t)$.

Bayes Filter

- An efficient way to compute beliefs from measurement and control data.

Bayes filter algorithm	
$bel(x_0) = p(x_0)$	
For $t = 1, \dots$	
$\quad \overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1})\, bel(x_{t-1})\, dx_{t-1}$	// Prediction
$\quad bel(x_t) = \eta\, p(z_t \mid x_t)\, \overline{bel}(x_t)$	// Correction

Bayes Filter

- An efficient way to compute beliefs from measurement and control data.

Bayes filter algorithm		
1	$bel(x_0) = p(x_0)$	
2	For $t = 1, \ldots$	
3	$\overline{bel}(x_t) = \int p(x_t \vert u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}$	// Prediction
4	$bel(x_t) = \eta p(z_t \vert x_t) \overline{bel}(x_t)$	// Correction

- In line 3, note that

$$\begin{aligned}\overline{bel}(x_t) &= \int p(x_t|x_{t-1}, z_{1:t-1}, u_{1:t}) p(x_{t-1}|z_{1:t-1}, u_{1:t}) dx_{t-1} \\ &= \int p(x_t|x_{t-1}, u_t) p(x_{t-1}|z_{1:t-1}, u_{1:t-1}) dx_{t-1} \\ &= \int p(x_t|u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}\end{aligned}$$

by the Markovian assumption and $x_{t-1} \perp_p u_t | z_{1:t-1}$.

Bayes Filter

- An efficient way to compute beliefs from measurement and control data.

Bayes filter algorithm	
1 $bel(x_0) = p(x_0)$	
2 For $t = 1, \ldots$	
3 $\quad \overline{bel}(x_t) = \int p(x_t \vert u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}$	// Prediction
4 $\quad bel(x_t) = \eta p(z_t \vert x_t) \overline{bel}(x_t)$	// Correction

- In line 3, note that

$$\begin{aligned}\overline{bel}(x_t) &= \int p(x_t|x_{t-1}, z_{1:t-1}, u_{1:t}) p(x_{t-1}|z_{1:t-1}, u_{1:t}) dx_{t-1} \\ &= \int p(x_t|x_{t-1}, u_t) p(x_{t-1}|z_{1:t-1}, u_{1:t-1}) dx_{t-1} \\ &= \int p(x_t|u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}\end{aligned}$$

  by the Markovian assumption and $x_{t-1} \perp_p u_t | z_{1:t-1}$.

- In line 4, note that

$$bel(x_t) \propto p(z_t|x_t, z_{1:t-1}, u_{1:t}) p(x_t|z_{1:t-1}, u_{1:t}) = p(z_t|x_t) \overline{bel}(x_t)$$

  by the Markovian assumption.

Bayes Filter

- An efficient way to compute beliefs from measurement and control data.

Bayes filter algorithm	
1 $bel(x_0) = p(x_0)$	
2 For $t = 1, \ldots$	
3 $\quad \overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}$	// Prediction
4 $\quad bel(x_t) = \eta p(z_t \mid x_t) \overline{bel}(x_t)$	// Correction

- In line 3, note that

$$\begin{aligned}\overline{bel}(x_t) &= \int p(x_t \mid x_{t-1}, z_{1:t-1}, u_{1:t}) p(x_{t-1} \mid z_{1:t-1}, u_{1:t}) dx_{t-1} \\ &= \int p(x_t \mid x_{t-1}, u_t) p(x_{t-1} \mid z_{1:t-1}, u_{1:t-1}) dx_{t-1} \\ &= \int p(x_t \mid u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}\end{aligned}$$

  by the Markovian assumption and $x_{t-1} \perp_p u_t \mid z_{1:t-1}$.

- In line 4, note that

$$bel(x_t) \propto p(z_t \mid x_t, z_{1:t-1}, u_{1:t}) p(x_t \mid z_{1:t-1}, u_{1:t}) = p(z_t \mid x_t) \overline{bel}(x_t)$$

  by the Markovian assumption.

- The algorithm is efficient only if lines 3-4 can be evaluated in closed form.

Kalman Filter

- Kalman filter $=$ Bayes filter $+$ linear Gaussian SSMs $\Rightarrow$ efficient.

Kalman Filter

- Kalman filter = Bayes filter + linear Gaussian SSMs ⇒ efficient.
- Recall that a linear Gaussian SSM is defined as

$$x_t = A_t x_{t-1} + B_t u_t + \epsilon_t$$
$$z_t = C_t x_t + \delta_t$$
$$x_0 = \mu_0 + \tau_0$$

where

$$\epsilon_t \sim \mathcal{N}(\epsilon_t | 0, R_t)$$
$$\delta_t \sim \mathcal{N}(\delta_t | 0, Q_t)$$
$$\tau_0 \sim \mathcal{N}(\tau_0 | 0, \Sigma_0)$$

Kalman Filter

- Kalman filter = Bayes filter + linear Gaussian SSMs ⇒ efficient.
- Recall that a linear Gaussian SSM is defined as

$$
\begin{aligned}
x_t &= A_t x_{t-1} + B_t u_t + \epsilon_t \\
z_t &= C_t x_t + \delta_t \\
x_0 &= \mu_0 + \tau_0
\end{aligned}
$$

where

$$
\begin{aligned}
\epsilon_t &\sim \mathcal{N}(\epsilon_t | 0, R_t) \\
\delta_t &\sim \mathcal{N}(\delta_t | 0, Q_t) \\
\tau_0 &\sim \mathcal{N}(\tau_0 | 0, \Sigma_0)
\end{aligned}
$$

or equivalently

$$
\begin{aligned}
p(x_t | x_{t-1}, u_t) &= \mathcal{N}(x_t | A_t x_{t-1} + B_t u_t, R_t) \\
p(z_t | x_t) &= \mathcal{N}(z_t | C_t x_t, Q_t) \\
p(x_0) &= \mathcal{N}(x_0 | \mu_0, \Sigma_0)
\end{aligned}
$$

Kalman Filter

- Kalman filter = Bayes filter + linear Gaussian SSMs $\Rightarrow$ efficient.
- Recall that a linear Gaussian SSM is defined as

$$
\begin{aligned}
x_t &= A_t x_{t-1} + B_t u_t + \epsilon_t \\
z_t &= C_t x_t + \delta_t \\
x_0 &= \mu_0 + \tau_0
\end{aligned}
$$

where

$$
\begin{aligned}
\epsilon_t &\sim \mathcal{N}(\epsilon_t | 0, R_t) \\
\delta_t &\sim \mathcal{N}(\delta_t | 0, Q_t) \\
\tau_0 &\sim \mathcal{N}(\tau_0 | 0, \Sigma_0)
\end{aligned}
$$

or equivalently

$$
\begin{aligned}
p(x_t | x_{t-1}, u_t) &= \mathcal{N}(x_t | A_t x_{t-1} + B_t u_t, R_t) \\
p(z_t | x_t) &= \mathcal{N}(z_t | C_t x_t, Q_t) \\
p(x_0) &= \mathcal{N}(x_0 | \mu_0, \Sigma_0)
\end{aligned}
$$

- Note that x_t, z_t and u_t may be vectors (of different dimensions).

Kalman Filter

- Kalman filter = Bayes filter + linear Gaussian SSMs $\Rightarrow$ efficient.
- Recall that a linear Gaussian SSM is defined as

$$\begin{aligned} x_t &= A_t x_{t-1} + B_t u_t + \epsilon_t \\ z_t &= C_t x_t + \delta_t \\ x_0 &= \mu_0 + \tau_0 \end{aligned}$$

  where

$$\begin{aligned} \epsilon_t &\sim \mathcal{N}(\epsilon_t | 0, R_t) \\ \delta_t &\sim \mathcal{N}(\delta_t | 0, Q_t) \\ \tau_0 &\sim \mathcal{N}(\tau_0 | 0, \Sigma_0) \end{aligned}$$

  or equivalently

$$\begin{aligned} p(x_t | x_{t-1}, u_t) &= \mathcal{N}(x_t | A_t x_{t-1} + B_t u_t, R_t) \\ p(z_t | x_t) &= \mathcal{N}(z_t | C_t x_t, Q_t) \\ p(x_0) &= \mathcal{N}(x_0 | \mu_0, \Sigma_0) \end{aligned}$$

- Note that x_t, z_t and u_t may be vectors (of different dimensions).
- Note that R_t and Q_t depend on t, i.e. no stationarity assumption. Fine for reasoning but not so fine for learning.

Kalman Filter

- $\overline{bel}(x_t) = \mathcal{N}(x_t | \overline{\mu}_t, \overline{\Sigma}_t)$
- $bel(x_t) = \mathcal{N}(x_t | \mu_t, \Sigma_t)$

Kalman filter algorithm

Set μ_0 and Σ_0 from $p(x_0)$
For $t = 1, \ldots$
  $\overline{\mu}_t = A_t \mu_{t-1} + B_t u_t$ // Prediction
  $\overline{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$ // Prediction
  $K_t = \overline{\Sigma}_t C_t^T (C_t \overline{\Sigma}_t C_t^T + Q_t)^{-1}$ // Kalman gain
  $\mu_t = \overline{\mu}_t + K_t (z_t - C_t \overline{\mu}_t)$ // Correction
  $\Sigma_t = (I - K_t C_t) \overline{\Sigma}_t$ // Correction

---

Kalman Filter

- $\overline{bel}(x_t) = \mathcal{N}(x_t | \overline{\mu}_t, \overline{\Sigma}_t)$
- $bel(x_t) = \mathcal{N}(x_t | \mu_t, \Sigma_t)$

Kalman filter algorithm

---

1 Set μ_0 and Σ_0 from $p(x_0)$
2 For $t = 1, \ldots$
3 $\quad \overline{\mu}_t = A_t \mu_{t-1} + B_t u_t$ // Prediction
4 $\quad \overline{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$ // Prediction
5 $\quad K_t = \overline{\Sigma}_t C_t^T (C_t \overline{\Sigma}_t C_t^T + Q_t)^{-1}$ // Kalman gain
6 $\quad \mu_t = \overline{\mu}_t + K_t (z_t - C_t \overline{\mu}_t)$ // Correction
7 $\quad \Sigma_t = (I - K_t C_t) \overline{\Sigma}_t$ // Correction

---

- Note that $(z_t - C_t \overline{\mu}_t)$ in line 6 is the deviation between the actual measurement and the predicted measurement.
- The Kalman gain K_t specifies the impact of the deviation in the correction.

Kalman Filter

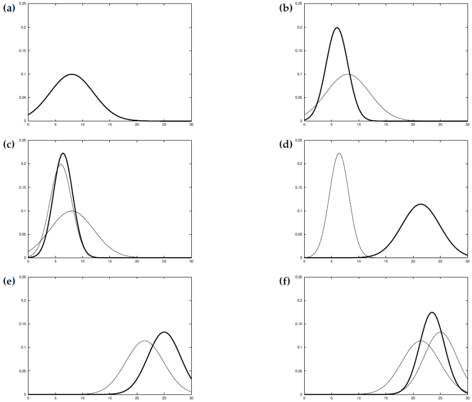


Figure 3.2 Illustration of Kalman filters: (a) initial belief, (b) a measurement (in bold) with the associated uncertainty, (c) belief after integrating the measurement into the belief using the Kalman filter algorithm, (d) belief after motion to the right (which introduces uncertainty), (e) a new measurement with associated uncertainty, and (f) the resulting belief.

- (a) $\overline{bel}(x_t)$, (b) $p(z_t|x_t)$, (c) $bel(x_t)$.
- (d) $\overline{bel}(x_{t+1})$, (e) $p(z_{t+1}|x_{t+1})$, (f) $bel(x_{t+1})$.

Kalman Filter

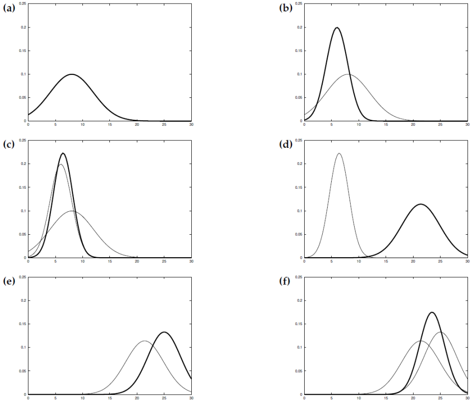


Figure 3.2 Illustration of Kalman filters: (a) initial belief, (b) a measurement (in bold) with the associated uncertainty, (c) belief after integrating the measurement into the belief using the Kalman filter algorithm, (d) belief after motion to the right (which introduces uncertainty), (e) a new measurement with associated uncertainty, and (f) the resulting belief.

- (a) $\overline{bel}(x_t)$, (b) $p(z_t|x_t)$, (c) $bel(x_t)$.
- (d) $\overline{bel}(x_{t+1})$, (e) $p(z_{t+1}|x_{t+1})$, (f) $bel(x_{t+1})$.
- Prediction increases uncertainty, and correction decreases it.

Kalman Filter

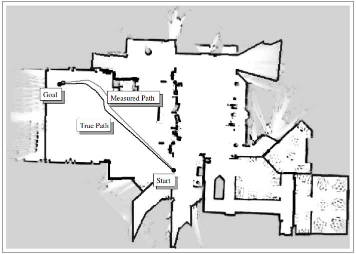
- Gaussian distributions are unimodal. Hence, $bel(x_t)$ is unimodal.
- This fits many robot localization scenarios but not all, e.g. the door scenario where the robot may be in several distant locations.

Kalman Filter

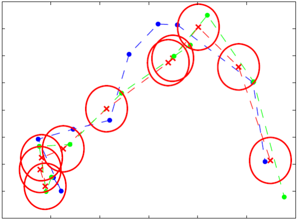
- Gaussian distributions are unimodal. Hence, $bel(x_t)$ is unimodal.
- This fits many robot localization scenarios but not all, e.g. the door scenario where the robot may be in several distant locations.
- Trade-off between accuracy and simplicity/tractability/efficiency.

Kalman Filter

- Gaussian distributions are unimodal. Hence, $bel(x_t)$ is unimodal.
- This fits many robot localization scenarios but not all, e.g. the door scenario where the robot may be in several distant locations.
- Trade-off between accuracy and simplicity/tractability/efficiency.

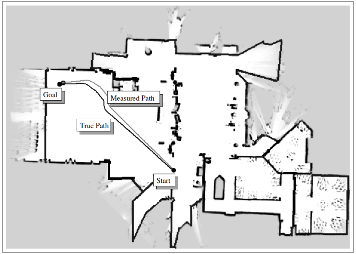


An illustration of a linear dynamical system being used to track a moving object. The blue points indicate the true positions of the object in a two-dimensional space at successive time steps, the green points denote noisy measurements of the positions, and the red crosses indicate the means of the inferred posterior distributions of the positions obtained by running the Kalman filtering equations. The covariances of the inferred positions are indicated by the red ellipses, which correspond to contours having one standard deviation.

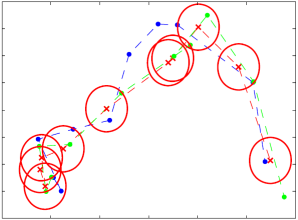


Kalman Filter

- Gaussian distributions are unimodal. Hence, $bel(x_t)$ is unimodal.
- This fits many robot localization scenarios but not all, e.g. the door scenario where the robot may be in several distant locations.
- Trade-off between accuracy and simplicity/tractability/efficiency.



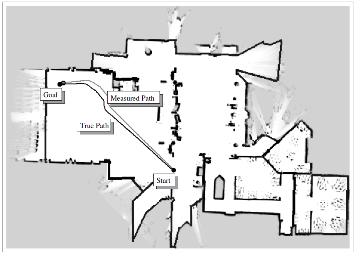
An illustration of a linear dynamical system being used to track a moving object. The blue points indicate the true positions of the object in a two-dimensional space at successive time steps, the green points denote noisy measurements of the positions, and the red crosses indicate the means of the inferred posterior distributions of the positions obtained by running the Kalman filtering equations. The covariances of the inferred positions are indicated by the red ellipses, which correspond to contours having one standard deviation.



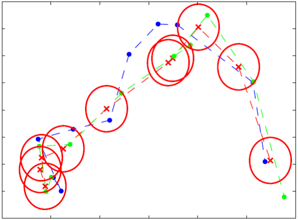
- Other examples:
  - No visual contact of the robot, e.g. on Mars and u_t = Earth's commands.

Kalman Filter

- Gaussian distributions are unimodal. Hence, $bel(x_t)$ is unimodal.
- This fits many robot localization scenarios but not all, e.g. the door scenario where the robot may be in several distant locations.
- Trade-off between accuracy and simplicity/tractability/efficiency.



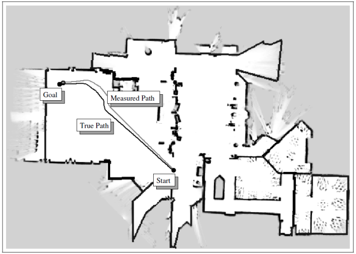
An illustration of a linear dynamical system being used to track a moving object. The blue points indicate the true positions of the object in a two-dimensional space at successive time steps, the green points denote noisy measurements of the positions, and the red crosses indicate the means of the inferred posterior distributions of the positions obtained by running the Kalman filtering equations. The covariances of the inferred positions are indicated by the red ellipses, which correspond to contours having one standard deviation.



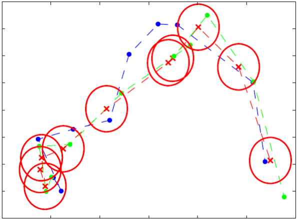
- Other examples:
  - No visual contact of the robot, e.g. on Mars and u_t = Earth's commands.
  - x_t = unemployment rate at time t, z_t = labor force survey, u_t = tax policy.

Kalman Filter

- Gaussian distributions are unimodal. Hence, $bel(x_t)$ is unimodal.
- This fits many robot localization scenarios but not all, e.g. the door scenario where the robot may be in several distant locations.
- Trade-off between accuracy and simplicity/tractability/efficiency.



An illustration of a linear dynamical system being used to track a moving object. The blue points indicate the true positions of the object in a two-dimensional space at successive time steps, the green points denote noisy measurements of the positions, and the red crosses indicate the means of the inferred posterior distributions of the positions obtained by running the Kalman filtering equations. The covariances of the inferred positions are indicated by the red ellipses, which correspond to contours having one standard deviation.



- Other examples:
  - No visual contact of the robot, e.g. on Mars and u_t = Earth's commands.
  - x_t = unemployment rate at time t, z_t = labor force survey, u_t = tax policy.
  - x_t = democrats' voting share at time t, z_t = poll result, u_t = tax policy.

Contents

- Linear Gaussian State Space Models
- Robot Localization
- Bayes Filter
- Kalman Filter

Thank you