Filtering and Robot Localization

Robert Platt
Northeastern University
Robot localization example

Goal: localize the robot based on sequential observations
- robot is given a map of the world; robot could be in any square
- initially, robot doesn't know which square it's in

Image: Berkeley CS188 course notes (downloaded Summer 2015)
Robot localization example

Robot perceives that there are walls above and below, but no walls either left or right

On each time step, the robot moves, and then observes the directions in which there are walls.
– observes a four-bit binary number
– observations are noisy: there is a small chance that each bit will be flipped.

Gray level denotes estimated probability that robot is in that square

Prob

0

1

Image: Berkeley CS188 course notes (downloaded Summer 2015)
Robot localization example

Image: Berkeley CS188 course notes (downloaded Summer 2015)
Robot localization example

Image: Berkeley CS188 course notes (downloaded Summer 2015)
Robot localization example

![Robot localization example diagram](Image: Berkeley CS188 course notes (downloaded Summer 2015))
Robot localization example

**Question**: how do we update this probability distribution from time $t$ to $t+1$?
State, $X_t$, is assumed to be unobserved.

However, you get to make one observation, $E_t$, on each timestep.
Hidden Markov Models (HMMs)

Process dynamics: $P(X_t | X_{t-1})$ — How the system changes from one time step to the next

Observation dynamics: $P(E_t | X_t)$ — What gets observed as a function of what state the system is in
Hidden Markov Models (HMMs)

Process dynamics: $P(X_t | X_{t-1})$  
How the system changes from one time step to the next

Observation dynamics: $P(E_t | X_t)$  
What gets observed as a function of what state the system is in

Let's assume (for now) that these probability distributions are given to us.
Hidden Markov Models (HMMs)

Process dynamics: \[ P(X_t|X_{t-1}) = P(X_t|X_{t-1}, \ldots, X_1) \]

Observation dynamics: \[ P(E_t|X_t) = P(E_t|X_t, X_{t-1}, \ldots, X_1) \]
Hidden Markov Models (HMMs)

Process dynamics: \[ P(X_t|X_{t-1}) = P(X_t|X_{t-1}, \ldots, X_1) \]

Observation dynamics: \[ P(E_t|X_t) = P(E_t|X_t, X_{t-1}, \ldots, X_1) \]

Markov assumptions
HMM example

Images: Berkeley CS188 course notes (downloaded Summer 2015)
Bayes Filtering

How do we go from this distribution to this distribution?
Bayes Filtering

\[ X_t \rightarrow X_{t+1} \rightarrow E_{t+1} \]
Bayes Filtering

\[ B(X_t) \]
\[ P(X_t|E_{1:t}) \]

\[ B'(X_t) \]
\[ P(X_{t+1}|E_{1:t}) \]

\[ B(X_{t+1}) \]
\[ P(X_{t+1}|E_{1:t+1}) \]

Process update
Observation update
Process update

\[ B(X_t) \quad \rightarrow \quad B'(X_t) \]

\[ P(X_t|e_{1:t}) \quad \rightarrow \quad P(X_{t+1}|e_{1:t}) \]
Process update

\[ P(X_t | e_{1:t}) \rightarrow P(X_{t+1} | e_{1:t}) \]

Different states from which \( x_{t+1} \) can be reached

\[ P(X_{t+1} | e_{1:t}) = \sum_{X_t} P(X_{t+1} | X_t, e_{1:t}) P(X_t | e_{1:t}) \]

Marginalize over next states
Process update

\[ B(X_t) \rightarrow B'(X_t) \]

Different states from which \( x_{t+1} \) can be reached

\[ B'(X_{t+1}) = \sum_{X_t} P(X_{t+1}|X_t, e_{1:t}) B(X_t) \]

Marginalize over next states
Process update

Before process update

After process update

\[ B'(X_{t+1}) = \sum_{X_t} P(X_{t+1} | X_t, e_{1:t}) B(X_t) \]

This is a little like convolution...

Each time you execute a process update, belief gets more disburst
– i.e. Shannon entropy increases
– this makes sense: as you predict state further into the future,
your uncertainty grows.
Bayes Filtering

\[ B(X_t) \]

\[ P(X_t | E_{1:t}) \] \[ \rightarrow \] \[ P(X_{t+1} | E_{1:t}) \] \[ \rightarrow \] \[ P(X_{t+1} | E_{1:t+1}) \]

Process update

Observation update
Observation update

\[ B'(X_t) \xrightarrow{\text{P}(X_{t+1}|e_{1:t})} B(X_{t+1}) \]

\[ P(X_{t+1}|e_{1:t+1}) = \eta P(e_{t+1}|X_{t+1}) P(X_{t+1}|e_{1:t}) \]
Observation update

\[
B'(X_t) \quad \rightarrow \quad B(X_{t+1})
\]

\[
P(X_{t+1}|e_{1:t}) \quad \rightarrow \quad P(X_{t+1}|e_{1:t+1})
\]

\[
P(X_{t+1}|e_{1:t+1}) = \eta P(e_{t+1}|X_{t+1})P(X_{t+1}|e_{1:t})
\]

Probability of seeing observation \(e_{t+1}\) from state \(X_{t+1}\)
Observation update

\[ B'(X_t) \rightarrow B(X_{t+1}) \]

\[ P(X_{t+1}|e_{1:t}) \quad \rightarrow \quad P(X_{t+1}|e_{1:t+1}) \]

\[ P(X_{t+1}|e_{1:t+1}) = \eta P(e_{t+1}|X_{t+1}) P(X_{t+1}|e_{1:t}) \]

\[ B(X_{t+1}) = \eta P(e_{t+1}|X_{t+1}) B'(X_{t+1}) \]

Where \( \eta = \frac{1}{P(e_{t+1})} \) is a normalization factor
Observation update

Before observation update

After observation update

\[ B(X_{t+1}) = \eta P(e_{t+1} | X_{t+1}) B'(X_{t+1}) \]
Weather HMM example

B(+r) = 0.5
B(-r) = 0.5

| $R_t$ | $R_{t+1}$ | $P(R_{t+1}|R_t)$ |
|-------|-----------|------------------|
| +r    | +r        | 0.7              |
| +r    | -r        | 0.3              |
| -r    | +r        | 0.3              |
| -r    | -r        | 0.7              |

| $R_t$ | $U_t$ | $P(U_t|R_t)$ |
|-------|-------|--------------|
| +r    | +u    | 0.9          |
| +r    | -u    | 0.1          |
| -r    | +u    | 0.2          |
| -r    | -u    | 0.8          |
Weather HMM example

\[ \begin{align*}
R_t & \rightarrow R_{t+1} \\
B'(+r) & = 0.5 \\
B'(-r) & = 0.5 \\
B(+r) & = 0.5 \\
B(-r) & = 0.5
\end{align*} \]

\[ \begin{array}{c|cc}
R_t & R_{t+1} & P(R_{t+1}|R_t) \\
\hline
+r & +r & 0.7 \\
+r & -r & 0.3 \\
-r & +r & 0.3 \\
-r & -r & 0.7
\end{array} \]

\[ \begin{array}{c|cc}
R_t & U_t & P(U_t|R_t) \\
\hline
+r & +u & 0.9 \\
+r & -u & 0.1 \\
-r & +u & 0.2 \\
-r & -u & 0.8
\end{array} \]

Slide: Berkeley CS188 course notes (downloaded Summer 2015)
Weather HMM example

\[ P(R_{t+1} | R_t) \]

\[ +r \quad +r \quad 0.7 \\
+u \quad -r \quad 0.3 \\
-u \quad +r \quad 0.3 \\
-r \quad -r \quad 0.7 \]

\[ B(+r) = 0.5 \]
\[ B'(+r) = 0.5 \]
\[ B(-r) = 0.5 \]
\[ B'(r) = 0.5 \]

\[ B(+r) = 0.818 \]
\[ B(-r) = 0.182 \]

\[ R_t \quad R_{t+1} \quad P(R_{t+1} | R_t) \]

\[ U_t \quad P(U_t | R_t) \]

\[ +r \quad +u \quad 0.9 \\
+u \quad -r \quad 0.1 \\
-r \quad +u \quad 0.2 \\
-r \quad -u \quad 0.8 \]
Weather HMM example

\[ P(R_{t+1} | R_t) \]

\[
\begin{array}{c|c|c}
R_t & R_{t+1} & P(R_{t+1} | R_t) \\
\hline
+r & +r & 0.7 \\
+r & -r & 0.3 \\
-r & +r & 0.3 \\
-r & -r & 0.7 \\
\end{array}
\]

\[
\begin{array}{c|c|c}
R_t & U_t & P(U_t | R_t) \\
\hline
+r & +u & 0.9 \\
+r & -u & 0.1 \\
-r & +u & 0.2 \\
-r & -u & 0.8 \\
\end{array}
\]

\[ B(\pm r) = 0.5 \]
\[ B'(\pm r) = 0.5 \]

\[ B(\pm r) = 0.818 \]
\[ B'(\pm r) = 0.627 \]

\[ B(-r) = 0.182 \]
\[ B'(-r) = 0.373 \]
Weather HMM example

\[ \begin{align*}
R_t & \quad R_{t+1} & \quad P(R_{t+1}|R_t) \\
+r & \quad +r & \quad 0.7 \\
+r & \quad -r & \quad 0.3 \\
-r & \quad +r & \quad 0.3 \\
-r & \quad -r & \quad 0.7 \\
\end{align*} \]

\[ \begin{align*}
B(+r) & = 0.5 \\
B(-r) & = 0.5 \\
B'(+r) & = 0.5 \\
B'(-r) & = 0.5 \\
\end{align*} \]

\[ \begin{align*}
B(+r) & = 0.818 \\
B(-r) & = 0.182 \\
B'(+r) & = 0.627 \\
B'(-r) & = 0.373 \\
\end{align*} \]

\[ \begin{align*}
B(+r) & = 0.883 \\
B(-r) & = 0.117 \\
B'(+r) & = 0.627 \\
B'(-r) & = 0.373 \\
\end{align*} \]

\[ \begin{align*}
R_t & \quad U_t & \quad P(U_t|R_t) \\
+r & \quad +u & \quad 0.9 \\
+r & \quad -u & \quad 0.1 \\
-r & \quad +u & \quad 0.2 \\
-r & \quad -u & \quad 0.8 \\
\end{align*} \]
Robot localization example
Robot localization example

Slide: Berkeley CS188 course notes (downloaded Summer 2015)
Robot localization example
Robot localization example
Robot localization example
Robot localization example

Slide: Berkeley CS188 course notes (downloaded Summer 2015)
Robot localization example

![Diagram of robot localization](image)

Prob

0 1

Slide: Berkeley CS188 course notes (downloaded Summer 2015)
Robot localization example
Robot localization example
Robot localization example
Applications of HMMs

- **Speech recognition HMMs:**
  - Observations are acoustic signals (continuous valued)
  - States are specific positions in specific words (so, tens of thousands)

- **Machine translation HMMs:**
  - Observations are words (tens of thousands)
  - States are translation options

- **Robot tracking:**
  - Observations are range readings (continuous)
  - States are positions on a map (continuous)
Particle Filter

Why must I be confined to this grid?

Standard Bayes filtering requires discretizing state space into grid cells.

Can do Bayes filtering w/o discretizing?
   – yes: particle filtering or Kalman filtering
Sequential Bayes Filtering is great, but it's not great for continuous state spaces.  
– you need to discretize the state space (e.g. a grid) in order to use Bayes filtering  
– but, doing filtering on a grid is not efficient...  

Therefore:  
– particle filters  
– Kalman filters  

Two different ways of filtering in continuous state spaces
Key idea: represent a probability distribution as a finite set of points

- density of points encodes probability mass.
- particle filtering is an adaptation of Bayes filtering to this particle representation
Suppose you are given an unknown probability distribution, \( P(x) \)

Suppose you can't evaluate the distribution analytically, but you can draw samples from it.

What can you do with this information?

\[
E_{x \sim P(x)}(f(x)) = \int_x f(x)P(x) \\
\approx \frac{1}{k} \sum_{i=1}^k f(x^i) \quad \text{where } x^i \text{ are samples drawn from } P(x)
\]
Monte Carlo Sampling

Suppose you are given an unknown probability distribution, \( P(x) \)

Suppose you can't evaluate the distribution analytically, but you can draw samples from it

What can you do with this information?

\[
E_{x \sim P(x)}(h(x)) = \int_x h(x) P(x)
\]

\[
\approx \frac{1}{k} \sum_{i=1}^{k} h(x^i) \quad \text{where } x^i \text{ are samples drawn from } P(x)
\]

FYI:
You can use the same strategy to estimate other moments as well...
Suppose you are given an unknown probability distribution, \( P(x) \).

Suppose you can't evaluate the distribution analytically, but you can draw samples from it.

What can you do with this information?

Suppose you can't even sample from it?

Suppose that all you can do is evaluate the function at a given point?
Importance Sampling

Question: how estimate expected values if cannot draw samples from f(x) - suppose all we can do is evaluate f(x) at a given point...
Importance Sampling

Question: how estimate expected values if cannot draw samples from $f(x)$
– suppose all we can do is evaluate $f(x)$ at a given point...

Answer: draw samples from a different distribution and weight them

Image: Thrun CS223b Course Notes (downloaded Summer 2015)
Importance Sampling

Question: how estimate expected values if cannot draw samples from f(x)  
– suppose all we can do is evaluate f(x) at a given point...

$E_{x \sim f(x)}(h(x)) = \int_x h(x) \frac{f(x)}{g(x)} g(x)$

$\approx \frac{1}{k} \sum_{i=1}^{k} h(x^i)w_i$  
where $x^i$ are samples drawn from $g(x)$

and $w_i = \frac{f(x^i)}{g(x^i)}$

Answer: draw samples from a different distribution and weight them

Image: Thrun CS223b Course Notes (downloaded Summer 2015)
Particle Filter

Prior distribution

\[ x^1_t, \ldots, x^n_t \quad w^1_t, \ldots, w^n_t = 1 \]
Particle Filter

Prior distribution

\[
\begin{align*}
    x_t^1, \ldots, x_t^n & \quad w_t^1, \ldots, w_t^n = 1
\end{align*}
\]

Process update

\[
\bar{x}_{t+1}^i \sim P(X_{t+1} | x_t^i, e_{1:t})
\]
Particle Filter

Prior distribution
\[ x_t^1, \ldots, x_t^n \quad w_t^1, \ldots, w_t^n = 1 \]

Process update
\[ \bar{x}_{t+1}^i \sim P(X_{t+1} \mid x_t^i, e_{1:t}) \]

Observation update
\[ w_{t+1}^i = P(e_{t+1} \mid \bar{x}_{t+1}^i) w_t^i \]
Particle Filter

\( B(X_t) \) \hspace{1cm} P(X_t|E_{1:t})

\( B'(X_t) \) \hspace{1cm} P(X_{t+1}|E_{1:t})

\( B(X_{t+1}) \) \hspace{1cm} P(X_{t+1}|E_{1:t+1})

Prior distribution

\( x_t^1, \ldots, x_t^n \quad w_t^1, \ldots, w_t^n = 1 \)

Process update

\( \bar{x}_{t+1}^i \sim P(X_{t+1}|x_t^i, e_{1:t}) \)

Observation update

\( w_{t+1}^i = P(e_{t+1}|\bar{x}_{t+1}^i)w_t^i \)

Resample

\( X_{t+1} = \{} \)

\( X_{t+1} = X_{t+1} \cup \bar{x}_{t+1}^i \) w/ prob \( w_{t+1}^i \)

Do this \( n \) times
Particle Filter

Prior distribution
Particle Filter

Prior distribution

Measurement update
Particle Filter

Process update

Resampling
Particle Filter

Measurement update
Particle Filter

Process update

Measurement update
Particle Filter Example
Particle Filter Example
## Particle Filtering

<table>
<thead>
<tr>
<th>Pros:</th>
<th>Cons:</th>
</tr>
</thead>
<tbody>
<tr>
<td>– works in continuous spaces</td>
<td>– parameters to tune</td>
</tr>
<tr>
<td>– can represent multi-modal distributions</td>
<td>– sample impoverishment</td>
</tr>
</tbody>
</table>
### Sample Impoverishment

<table>
<thead>
<tr>
<th><strong>Pros:</strong></th>
<th><strong>Cons:</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>– works in continuous spaces</td>
<td>– parameters to tune</td>
</tr>
<tr>
<td>– can represent multi-modal distributions</td>
<td>– sample impoverishment</td>
</tr>
</tbody>
</table>

No particles nearby the true system state
Sample Impoverishment

Prior distribution
\[ x_t^n, w_t^1, \ldots, w_t^n = 1 \]

Process update
\[ P(X_{t+1} | x_t^i, e_{1:t}) \]

Observation update
\[ \omega_{t+1} = P(e_{t+1} | \bar{x}_{t+1}^i) w_t^i \]

Resample
\[ X_{t+1} = \emptyset \]
\[ X_{t+1} = X_{t+1} \cup \bar{x}_{t+1}^i \text{ w/ prob } w_{t+1}^i \]

If there aren't enough samples, then we might ``resample away'' the true state...

Do this \( n \) times
Sample Impoverishment

If there aren't enough samples, then we might ``resample away" the true state...

**One solution**: add an additional $k$ samples drawn completely at random.

Do this $n$ times.

---

**Prior distribution**

$$x_t^n, \ldots, w_t^1, \ldots, w_t^n = 1$$

**Process update**

$$P(X_{t+1} \mid x_t^i, e_1:t)$$

**Observation update**

$$P(e_{t+1} \mid \bar{x}_{t+1}^i)w_t^i$$

**Resample**

$$X_{t+1} = \emptyset$$

$$X_{t+1} = X_{t+1} \cup \bar{x}_{t+1}^i \text{ w/ prob } w_{t+1}^i$$
If there aren't enough samples, then we might ``resample away'' the true state...

One solution: add an additional $k$ samples drawn completely at random

BUT: there's always a chance that the true state won't be represented well by the particles...

$$B(X_{t+1}) \quad P(X_{t+1}|E_{1:t+1})$$

Do this $n$ times

$$X_{t+1} = \{ \}$$

$$X_{t+1} = X_{t+1} \cup \bar{x}^i_{t+1} \quad \text{w/ prob } w^i_{t+1}$$
Another way to adapt Sequential Bayes Filtering to continuous state spaces

– relies on representing the probability distribution as a Gaussian

– first developed in the early 1960s (before general Bayes filtering); used in Apollo program
Kalman Idea

initial position  prediction  measurement  update

Image: Thrun et al., CS233B course notes
Kalman Idea

\[ P(x_{t+1} | z_{0:t}) = \sum_{x_t} P(x_{t+1} | x_t) P(x_t | z_{0:t}) \]

\[ P(x_{t+1} | z_{0:t+1}) = \eta P(z_{t+1} | x_{t+1}) P(x_{t+1} | z_{0:t}) \]
Gaussians

- Univariate Gaussian:
  \[ P(x) = \eta e^{-\frac{1}{2} \frac{(x-\mu)^2}{\sigma^2}} \]

- Multivariate Gaussian:
  \[ P(x) = \eta e^{-\frac{1}{2} (x-\mu)^T \Sigma^{-1} (x-\mu)} \]
  \[ P(x) = N(x; \mu, \Sigma) \]
Playing w/ Gaussians

- Suppose: \( P(x) = N(x; \mu, \Sigma) \)
  \[ y = Ax + b \]

- Calculate: \( P(y) = ? \)
  \[ P(y) = N(y; Ax + b, A\Sigma A^T) \]

\[ x^T x = 1 \]

\[ (Ax + b)^T (Ax + b) = 1 \]
In fact

• Suppose: \( P(x) = N(x; \mu, \Sigma) \)
  \[ y = Ax + b \]

• Then:

\[
P \left( \begin{pmatrix} x \\ y \end{pmatrix} \right) = N \left[ \begin{pmatrix} x \\ y \end{pmatrix} : \begin{pmatrix} \mu \\ A\mu + b \end{pmatrix} , \begin{pmatrix} \Sigma & \Sigma A^T \\ A\Sigma & A\Sigma A^T \end{pmatrix} \right]
\]
Illustration

Image: Thrun et al., CS233B course notes
And

Suppose: \[ P(x) = N(x; \mu, \Sigma) \]
\[ P(y|x) = N(y; Ax + b, R) \]

Then:
\[
P \left( \begin{pmatrix} x \\ y \end{pmatrix} \right) = N \left[ \begin{pmatrix} x \\ y \end{pmatrix} : \begin{pmatrix} \mu \\ A\mu + b \end{pmatrix}, \begin{pmatrix} \Sigma & \Sigma A^T \\ A\Sigma & A\Sigma A^T + R \end{pmatrix} \right]
\]
\[ P(y) = N(y; A\mu + b, A\Sigma A^T + R) \]

Marginal distribution
Does this remind us of anything?
Does this remind us of anything?

Process update (discrete): 
\[ P(x_{t+1} | z_{0:t}) = \sum_{x_t} P(x_{t+1} | x_t) P(x_t | z_{0:t}) \]

Process update (continuous): 
\[ P(x_{t+1} | z_{0:t}) = \int_{x_t} P(x_{t+1} | x_t) P(x_t | z_{0:t}) \]
Does this remind us of anything?

**Process update (discrete):**

\[ P(x_{t+1} | z_{0:t}) = \sum_{x_t} P(x_{t+1} | x_t) P(x_t | z_{0:t}) \]

**Process update (continuous):**

\[ P(x_{t+1} | z_{0:t}) = \int_{x_t} P(x_{t+1} | x_t) P(x_t | z_{0:t}) \]

\[
N(x_{t+1} | Ax_t, Q) \\
\text{transition dynamics}
\]

\[
N(x_t | \mu_t, \Sigma_t) \\
\text{prior}
\]
Does this remind us of anything?

Process update (discrete): 
\[ P(x_{t+1} | z_{0:t}) = \sum_{x_t} P(x_{t+1} | x_t) P(x_t | z_{0:t}) \]

Process update (continuous): 
\[ P(x_{t+1} | z_{0:t}) = \int_{x_t} P(x_{t+1} | x_t) P(x_t | z_{0:t}) \]

transition dynamics \[ N(x_{t+1} | Ax_t, Q) \]

prior \[ N(x_t | \mu_t, \Sigma_t) \]

\[ P(x_{t+1} | z_{0:t}) = \int_{x_t} N(x_{t+1} | Ax_t, Q) N(x_t; \mu_t, \Sigma_t) \]

\[ P(x_{t+1} | z_{0:t}) = N(x_{t+1} | A\mu_t, A\Sigma_t A^T + Q) \]
Observation update

\[ P(x_{t+1} | z_{0:t+1}) = \eta P(z_{t+1} | x_{t+1}) P(x_{t+1} | z_{0:t}) \]

\[ N(z_{t+1} | C x_{t+1}, R) \]

\[ N(x_t | \mu'_t, \Sigma'_t) \]

Where:

\[ \mu'_t = A \mu_t \]

\[ \Sigma'_t = A \Sigma_t A^T + Q \]
Observation update

\[ P(x_{t+1}|z_{0:t+1}) = \eta P(z_{t+1}|x_{t+1})P(x_{t+1}|z_{0:t}) \]

\[ N(z_{t+1}|Cx_{t+1}, R) \quad N(x_t|\mu_t', \Sigma_t') \]

Where:\n\[ \mu_t' = A\mu_t \]
\[ \Sigma_t' = A\Sigma_t A^T + Q \]

\[ P(z_{t+1}, x_{t+1}|z_{0:t}) = \eta N(z_{t+1}|Cx_t, R)N(x_t; \mu_t', \Sigma_t') \]
Observation update

\[
P(x_{t+1} | z_{0:t+1}) = \eta P(z_{t+1} | x_{t+1}) P(x_{t+1} | z_{0:t})
\]

\[
N(z_{t+1} | Cx_{t+1}, R) \quad N(x_t | \mu'_t, \Sigma'_t)
\]

Where:
\[
\mu'_t = A \mu_t \\
\Sigma'_t = A \Sigma_t A^T + Q
\]

\[
P(z_{t+1}, x_{t+1} | z_{0:t}) = \eta N(z_{t+1} | Cx_t, R) N(x_t; \mu'_t, \Sigma'_t)
\]

\[
P(z_{t+1}, x_{t+1} | z_{0:t}) = N \left[ \begin{array}{c} x_{t+1} \\ z_{t+1} \end{array} : \begin{array}{c} \mu'_t \\ C \mu'_t \end{array} , \begin{pmatrix} \Sigma'_t & \Sigma'_t C^T \\ C \Sigma'_t & C \Sigma'_t A^T + R \end{pmatrix} \right]
\]
Observation update

\[ P(z_{t+1}, x_{t+1} | z_{0:t}) = N \left[ \begin{array}{c} x_{t+1} \\ z_{t+1} \end{array} : \begin{array}{l} \mu'_t \\ C \mu'_t \end{array}, \left( \begin{array}{cc} \Sigma'_t & \Sigma'_t C^T \\ C \Sigma'_t & C \Sigma'_t A^T + R \end{array} \right) \right] \]

But we need: \[ P(x_{t+1} | z_{0:t+t}) = ? \]
Another Gaussian identity...

Suppose: $N \left[ \begin{array}{c} x \\ y \end{array} : a \\ b \right] \left( \begin{array}{cc} A & C \\ C^T & B \end{array} \right)$

Calculate: $P(y|x) =$?

$$P(y|x) = N(y|b + C^T A^{-1} (x - a), B - C^T A^{-1} C)$$
Observation update

\[ P(z_{t+1}, x_{t+1}|z_{0:t}) = N \left[ \begin{array}{c} x_{t+1} \\ z_{t+1} \end{array} : \begin{array}{c} \mu'_t \\ C \mu'_t \end{array} , \begin{pmatrix} \Sigma & \Sigma C^T \\ C \Sigma & C \Sigma A^T + R \end{pmatrix} \right] \]

But we need: \( P(x_{t+1}|z_{0:t+1}) = ? \)

\[ P(x_{t+1}|z_{0:t+1}) = N(x_{t+1}; \mu_{t+1}, \Sigma_{t+1}) \]

\[
\begin{align*}
\mu_{t+1} &= \mu'_t + \Sigma'_t C^T (R + C \Sigma'_t C^T)^{-1} (z_{t+1} - C \mu'_t) \\
\Sigma_{t+1} &= \Sigma'_t - \Sigma'_t C^T (R + C \Sigma'_t C^T)^{-1} C \Sigma'_t
\end{align*}
\]
To summarize the Kalman filter

System:

\[ P(x_{t+1}|x_t) = N(x_{t+1}|Ax_t, Q) \]
\[ P(z_{t+1}|x_{t+1}) = N(z_{t+1}|Cx_{t+1}, R) \]

Prior: \( \mu_t \)
\[ \Sigma_t \]

Process update: \( \mu'_t = A\mu_t \)
\[ \Sigma'_t = A\Sigma_tA^T + Q \]

Measurement update:
\[ \mu_{t+1} = \mu'_t + \Sigma'_t C^T (R + C\Sigma'_t C^T)^{-1} (z_{t+1} - C\mu'_t) \]
\[ \Sigma_{t+1} = \Sigma'_t - \Sigma'_t C^T (R + C\Sigma'_t C^T)^{-1} C\Sigma'_t \]
Suppose there is an action term...

System: \[ P(x_{t+1} | x_t) = N(x_{t+1} | Ax_t + u_t, Q) \]
\[ P(z_{t+1} | x_{t+1}) = N(z_{t+1} | Cx_{t+1}, R) \]

Prior: \[ \mu_t \]
\[ \Sigma_t \]

Process update: \[ \mu'_t = A\mu_t + u_t \]
\[ \Sigma'_t = A\Sigma_t A^T + Q \]

Measurement update:
\[ \mu_{t+1} = \mu'_t + \Sigma'_t C^T (R + C\Sigma'_t C^T)^{-1} (z_{t+1} - C\mu'_t) \]
\[ \Sigma_{t+1} = \Sigma'_t - \Sigma'_t C^T (R + C\Sigma'_t C^T)^{-1} C\Sigma'_t \]
To summarize the Kalman filter

Prior: \( \mu_t \)
\[ \Sigma_t \]

Process update: \( \mu'_t = A \mu_t \)
\[ \Sigma'_t = A \Sigma_t A^T + Q \]

Measurement update:
\[ \mu_{t+1} = \mu'_t + \Sigma'_t C^T (R + C \Sigma_t C^T)^{-1} (z_{t+1} - C \mu'_t) \]

This factor is often called the “Kalman gain”

\[ \Sigma_{t+1} = \Sigma'_t - \Sigma'_t C^T (R + C \Sigma_t C^T)^{-1} C \Sigma'_t \]
Things to note about the Kalman filter

Process update: \[ \mu_t' = A\mu_t \quad \Sigma_t' = A\Sigma_t A^T + Q \]

Measurement update:
\[ \mu_{t+1} = \mu_t' + \Sigma_t' C^T (R + C\Sigma_t' C^T)^{-1} (z_{t+1} - C\mu_t') \]
\[ \Sigma_{t+1} = \Sigma_t' - \Sigma_t' C^T (R + C\Sigma_t' C^T)^{-1} C\Sigma_t' \]

– covariance update is independent of observation
– Kalman is only optimal for linear-Gaussian systems
– the distribution “stays” Gaussian through this update
– the error term can be thought of as the difference between the observation and the prediction
Kalman in 1D

System:
\[ P(x_{t+1}|x_t) = N(x_{t+1} : x_t + u_t, q) \]
\[ P(z_{t+1}|x_{t+1}) = N(z_{t+1}|2x_{t+1}, r) \]

Process update:
\[ \bar{\mu}_t = \mu_t + u_t \]
\[ \bar{\sigma}_t^2 = \sigma_t^2 + q \]

Measurement update:
\[ \mu_{t+1} = \bar{\mu}_t + \frac{2\bar{\sigma}_t^2}{r + 4\bar{\sigma}_t^2} (z_{t+1} - \bar{\mu}_t) \]
\[ \sigma_{t+1} = \bar{\sigma}_t^2 - \frac{4(\bar{\sigma}_t^2)^2}{r + 4\bar{\sigma}_t^2} \]
Kalman Idea

initial position  prediction  measurement  update

Image: Thrun et al., CS233B course notes
Example: estimate velocity

Image: Thrun et al., CS233B course notes
Example: filling a tank

\[ x = \begin{pmatrix} \ell \\ f \end{pmatrix} \quad \text{Level of tank, Fill rate} \]

\[ l_{t+1} = l_t + f \, dt \]

Process:

\[ x_{t+1} = \begin{pmatrix} 1 & dt \\ 0 & 1 \end{pmatrix} x_t + q \]

Observation:

\[ z_{t+1} = \begin{pmatrix} 1 & 0 \end{pmatrix} x_{t+1} + r \]
Example: estimate velocity

\[ x_{t+1} = Ax_t + w_t \]

\[
\begin{pmatrix}
  x_{t+1} \\
  y_{t+1} \\
  \dot{x}_{t+1} \\
  \dot{y}_{t+1}
\end{pmatrix}
= 
\begin{pmatrix}
  1 & 0 & dt & 0 \\
  0 & 1 & 0 & dt \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
  x_t \\
  y_t \\
  \dot{x}_t \\
  \dot{y}_t
\end{pmatrix}
+ w_t
\]

\[ z_{t+1} = Cx_{t+1} + r_{t+1} \]

\[
\begin{pmatrix}
  x_{t+1} \\
  y_{t+1} \\
  \dot{x}_{t+1} \\
  \dot{y}_{t+1}
\end{pmatrix}
= 
\begin{pmatrix}
  1 & 0 & 0 & 0 \\
  0 & 1 & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
  x_{t+1} \\
  y_{t+1} \\
  \dot{x}_{t+1} \\
  \dot{y}_{t+1}
\end{pmatrix}
+ r_{t+1}\]
But, my system is NON-LINEAR!

\[ x_{t+1} = f(x_t, u_t) \]
\[ \neq Ax_t + Bu_t \]

What should I do?
But, my system is NON-LINEAR!

\[ x_{t+1} = f(x_t, u_t) \]
\[ \neq Ax_t + Bu_t \]

Well, there are some options...

• What should I do?
But, my system is NON-LINEAR!

\[ x_{t+1} = f(x_t, u_t) \]

\[ \neq Ax_t + Bu_t \]

Well, there are some options...

- But none of them are great.

- What should I do?
But, my system is NON-LINEAR!

\[ x_{t+1} = f(x_t, u_t) \]
\[ \neq Ax_t + Bu_t \]

- What should I do?

Well, there are some options...

But none of them are great.

Here's one: the Extended Kalman Filter
Extended Kalman filter

Take a Taylor expansion:

\[ x_{t+1} = f(x_t, u_t) \]
\[ \approx f(\mu_t, u_t) + A_t(x_t - \mu_t) \]

Where:

\[ A_t = \frac{\partial f}{\partial x}(\mu_t, u_t) \]

\[ z_{t+1} = h(x_t) \]
\[ \approx h(\mu_t) + C_t(x_t - \mu_t) \]

Where:

\[ C_t = \frac{\partial h}{\partial x}(\mu_t) \]
Extended Kalman filter

Take a Taylor expansion:

\[ x_{t+1} = f(x_t, u_t) \]
\[ \approx f(\mu_t, u_t) + A_t(x_t - \mu_t) \]

Where:

\[ A_t = \frac{\partial f}{\partial x}(\mu_t, u_t) \]

\[ z_{t+1} = h(x_t) \]
\[ \approx h(\mu_t) + C_t(x_t - \mu_t) \]

Where:

\[ C_t = \frac{\partial h}{\partial x}(\mu_t) \]

Then use the same equations...
To summarize the EKF

Prior: $\mu_t$
$\Sigma_t$

Process update:
$\mu'_t = f(\mu_t, u_t)$
$\Sigma'_t = A_t \Sigma_t A^T_t + Q$

Measurement update:
$\mu_{t+1} = \mu'_t + \Sigma'_t C^T (R + \Sigma_t'C^T)^{-1} (z_{t+1} - h(\mu'_t))$
$\Sigma_{t+1} = \Sigma'_t - \Sigma'_t C^T (R + \Sigma'_t C^T)^{-1} C \Sigma'_t$
Extended Kalman filter

Image: Thrun et al., CS233B course notes
Extended Kalman filter

Image: Thrun et al., CS233B course notes
Each time you execute a process update, belief gets more dispersed — *i.e.* Shannon entropy increases — this makes sense: as you predict state further into the future, your uncertainty grows.

\[
B'(X_{t+1}) = \sum_{X_t} P(X_{t+1} | X_t, e_{1:t}) B(X_t)
\]

This is a little like convolution...

Images: Berkeley CS188 course notes (downloaded Summer 2015)
Kalman Filter
Observation update

Images: Berkeley CS188 course notes (downloaded Summer 2015)