EKF SLAM

D.A. Forsyth, UIUC
The SLAM Problem

- SLAM stands for simultaneous localization and mapping
- The task of building a map while estimating the pose of the robot relative to this map

- Why is SLAM hard?
  Chicken-or-egg problem:
  - a map is needed to localize the robot and a pose estimate is needed to build a map

From Burgard et al slides
Alternative view of SLAM

• We already know we can do it
  • for example
    • do the matrix factorization stuff incrementally
    • visual odometry then triangulate
  • BUT
    • that doesn’t take uncertainty into account

• What we’re doing now is
  • wrapping an EKF (other filter) around ideas we’ve seen before
Why is SLAM a hard problem?

- In the real world, the mapping between observations and landmarks is unknown
- Picking wrong data associations can have catastrophic consequences
- Pose error correlates data associations

From Burgard et al slides
In factorization language

- Which point in image i goes into which row of the matrix?
  - get that wrong enough often enough and you’re in trouble

- Obvious we can do something about this
  - eg assume we have OK reconstruction from frame 1..N-1
    - in frame N, estimate camera motion from
      - small number of reliable point correspondences + VO
      - shaft encoders, etc.
    - now sort out all other observations
      - eg map to the point that appears closest in predicted camera
Data Association Problem

- A data association is an assignment of observations to landmarks
- In general there are more than $\binom{n}{m}$ (n observations, m landmarks) possible associations
- Also called “assignment problem”

From Burgard et al slides
State

\[ \mathbf{x} = \begin{bmatrix} \mathcal{R} \\ \mathcal{M} \end{bmatrix} = \begin{bmatrix} \mathcal{R} \\ \mathcal{L}_1 \\ \vdots \\ \mathcal{L}_n \end{bmatrix} \]

Position and orientation of the robot

Landmark 1 position in OCF

All landmark positions in original coordinate frame
A general movement model

\[
\begin{bmatrix}
x \\
y \\
\theta
\end{bmatrix} \rightarrow \begin{bmatrix}
x + R(\sin(\theta + \Delta \theta) - \sin \theta) \\
y - R(\cos(\theta + \Delta \theta) - \cos \theta) \\
\theta + \Delta \theta
\end{bmatrix}
\]

THIS ISN’T LINEAR!

For sufficiently small timestep, bounded rate of change in angle, we get

\[
\begin{bmatrix}
x \\
y \\
\theta
\end{bmatrix} \rightarrow \begin{bmatrix}
x + v \cos \theta \\
y + v \sin \theta \\
\theta + u
\end{bmatrix}
\]

v, u parameters of motion

THIS ISN’T LINEAR!
A general movement model

\[
\begin{bmatrix}
  x \\
  y \\
  \theta
\end{bmatrix} 
\rightarrow 
\begin{bmatrix}
  x + R(\sin(\theta + \Delta \theta) - \sin \theta) \\
  y - R(\cos(\theta + \Delta \theta) - \cos \theta) \\
  \theta + \Delta \theta
\end{bmatrix}
\]

THIS ISN’T LINEAR!

\[ v_t = \text{velocity} \]
\[ \omega_t = \text{rotational velocity} \]

\[
\begin{bmatrix}
  x \\
  y \\
  \theta
\end{bmatrix} 
\rightarrow 
\begin{bmatrix}
  x \\
  y \\
  \theta
\end{bmatrix} + 
\begin{bmatrix}
  \frac{v_t}{\omega_t} (\sin(\theta + \omega_t \Delta t) - \sin(\theta)) \\
  -\frac{v_t}{\omega_t} (\cos(\theta + \omega_t \Delta t) - \cos(\theta)) \\
  \omega_t \Delta t
\end{bmatrix}
\]
Recall: The extended Kalman filter

- Linearize:

\[ \mathbf{x}_i = f(\mathbf{x}_{i-1}, \mathbf{n}) \]

\[ \mathcal{F}_x = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \cdots \\ \cdots & \frac{\partial f_i}{\partial x_j} & \cdots \\ \cdots & \cdots & \cdots \end{bmatrix} \]

\[ \mathcal{F}_n = \begin{bmatrix} \frac{\partial f_1}{\partial n_1} & \cdots & \cdots \\ \cdots & \frac{\partial f_i}{\partial n_j} & \cdots \\ \cdots & \cdots & \cdots \end{bmatrix} \]

Posterior covariance of \( \mathbf{x}_{\{i-1\}} \)

\[ \mathbf{x}_i \sim N(f(\overline{\mathbf{x}}_{i-1}^+, \mathbf{0}), \mathcal{F}_x \Sigma_{i-1}^{+} \mathcal{F}_x^T + \mathcal{F}_n \Sigma_{n,i} \mathcal{F}_n^T) \]

Noise covariance
Measuring position

- Landmark is at: $\begin{bmatrix} u \\ v \end{bmatrix}$
  - in world coordinate system
- We record position in vehicle’s frame:
  $$\begin{bmatrix} x_v \\ y_v \end{bmatrix} = \mathcal{R}_{-\theta} \begin{bmatrix} (u - x) \\ (v - y) \end{bmatrix}$$

Observation

THIS ISN’T LINEAR!
Recall: The extended Kalman filter

- Linearize:

\[ y_i = g(x_i, n) \]

\[ G_x = \begin{bmatrix} \frac{\partial g}{\partial x_1} & \cdots & \cdots \\ \cdots & \frac{\partial g}{\partial x_1} & \cdots \\ \cdots & \cdots & \cdots \end{bmatrix} \]

\[ G_n = \begin{bmatrix} \frac{\partial g}{\partial n_1} & \cdots & \cdots \\ \cdots & \frac{\partial g}{\partial n_1} & \cdots \end{bmatrix} \]

\[ y_i \approx \mathcal{N}(g(x_i, 0), G_x \Sigma_i G_x^T + G_n \Sigma_{m,i} G_n^T) \]
Dynamic Model:

\[
x_i = f(x_{i-1}, n)
\]

\[
y_i = g(x_i, n)
\]

Start Assumptions: \( \bar{x}_0^- \) and \( \Sigma_0^- \) are known

Update Equations: Prediction

\[
\begin{align*}
x_i &\sim \mathcal{N}(f(x_{i-1}, 0), \mathcal{F}_x \Sigma_{i-1}^+ \mathcal{F}_x^T + \mathcal{F}_n \Sigma_n, i \mathcal{F}_n^T) \\
\end{align*}
\]

Update Equations: Correction

\[
\begin{align*}
\mathcal{K}_i & = \Sigma_i^- \mathcal{M}_i^T \left[ \mathcal{G}_x \Sigma_i^- \mathcal{G}_x^T + \mathcal{G}_n \Sigma_m, i \mathcal{G}_n^T \right]^{-1} \\
\bar{x}_i^+ & = \bar{x}_i^- + \mathcal{K}_i [y_i - g(x_i^-, 0)] \\
\Sigma_i^+ & = [I - \mathcal{K}_i \mathcal{G}_x] \Sigma_i^-
\end{align*}
\]

? The extended kalman filter
Correction!

Dynamic Model:

\[ x_i = f(x_{i-1}, n) \]

\[ y_i = g(x_i, n) \]

Start Assumptions: \( \bar{x}_0^- \) and \( \Sigma_0^- \) are known

Update Equations: Prediction

\[ \bar{x}_i^- \sim \mathcal{N}(f(x_{i-1}, 0), \mathcal{F}_x \Sigma_{i-1}^+ \mathcal{F}_x^T + \mathcal{F}_n \Sigma_{n,i} \mathcal{F}_n^T) \]

Update Equations: Correction

\[ \mathcal{K}_i = \Sigma_i^- G_x^T \left[ G_x \Sigma_i^- G_x^T + G_n \Sigma_{m,i} G_n^T \right]^{-1} \]

\[ \bar{x}_i^+ = \bar{x}_i^- + \mathcal{K}_i [y_i - g(x_i^-, 0)] \]

\[ \Sigma_i^+ = [I - \mathcal{K}_i G_x] \Sigma_i^- \]

The extended kalman filter
In principle, now easy

• Rather horrid from the point of view of complexity
  • looks like we have to invert a 3+N by 3+N matrix!

• BUT
  • \( F_x \) is much simpler than it might look
    • the landmarks do not move!
  • \( F_n \) ditto
    • there is no noise in the landmark updates - the landmarks are fixed
  • Outcome:
    • We can deal with landmarks one by one
      • and so do many small matrix inversions rather than one large one
State update

\[ x_i = f(x_{i-1}, n) \]

\[ \begin{bmatrix}
    \mathcal{R} \\
    \mathcal{M} \\
    \mathcal{L}_1 \\
    \mathcal{L}_2 \\
    \ldots \\
    \mathcal{L}_n
\end{bmatrix} = \begin{bmatrix}
    \mathcal{R} \\
    \mathcal{L}_1 \\
    \ldots \\
    \mathcal{L}_n
\end{bmatrix} \]

- The vehicle moves, as above;
  - but the landmarks don’t move
  - and there isn’t any noise

\[
\begin{bmatrix}
    \mathcal{R} \\
    \mathcal{L}_1 \\
    \mathcal{L}_2 \\
    \ldots \\
    \mathcal{L}_n
\end{bmatrix} \rightarrow \begin{bmatrix}
    h(\mathcal{R}) + \xi \\
    \mathcal{L}_1 \\
    \mathcal{L}_2 \\
    \ldots \\
    \mathcal{L}_n
\end{bmatrix}
\]
In principle, now easy

- **BUT**
  - $F_x$ is much simpler than it might look
    - the landmarks do not move!
  - $F_n$ ditto
    - there is no noise in the landmark updates - the landmarks are fixed

$$
F_x = \begin{bmatrix}
3 & N \\
\frac{\partial f_R}{\partial R} & 0 \\
0 & \mathcal{I}
\end{bmatrix}
$$

$$
F_n = \begin{bmatrix}
\frac{\partial f_R}{\partial n} & 0 \\
0 & 0
\end{bmatrix}
$$

N=Number of landmarks
Effects:

- Imagine we have 2 landmarks

Recall EKF:

\[ x_i \sim \mathcal{N}(f(x_{i-1}, 0), \mathbf{F}_x \Sigma_{i-1}^+ \mathbf{F}_x^T + \mathbf{F}_n \Sigma_n, i \mathbf{F}_n^T) \]

\[
\mathbf{F}_x = \begin{bmatrix}
\mathbf{W} & 0 & 0 \\
0 & \mathcal{I} & 0 \\
0 & 0 & \mathcal{I}
\end{bmatrix}
\]

\[
\Sigma_{i-1}^+ = \begin{bmatrix}
\mathbf{A} & \mathbf{B} & \mathbf{C} \\
\mathbf{B}^T & \mathbf{D} & \mathbf{E} \\
\mathbf{C}^T & \mathbf{E}^T & \mathbf{F}
\end{bmatrix}
\]

\[
\mathbf{F}_x \Sigma_{i-1}^+ \mathbf{F}_x^T = \begin{bmatrix}
\mathbf{W} \mathbf{A} \mathbf{W}^T & \mathbf{W} \mathbf{A} & \mathbf{W} \mathbf{B} \\
\mathbf{B}^T \mathbf{W}^T & \mathbf{D} & \mathbf{E} \\
\mathbf{C}^T \mathbf{W} & \mathbf{E}^T & \mathbf{F}
\end{bmatrix}
\]

Notice fewer matrix multiplies!
Effects:

- Imagine we have 2 landmarks

Recall EKF:

\[
x_i \sim \mathcal{N}(f(x_{i-1}, 0), \mathcal{F}_x \Sigma_{i-1}^{+} \mathcal{F}_x^T + \mathcal{F}_n \Sigma_{n,i} \mathcal{F}_n^T)
\]

\[
\mathcal{F}_n = \begin{bmatrix}
\mathcal{V} & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}, \quad \Sigma_{n,i} = \begin{bmatrix}
\mathcal{H} & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\]

\[
\mathcal{F}_n \Sigma_{n,i} \mathcal{F}_n^T = \begin{bmatrix}
\mathcal{V} \mathcal{H} \mathcal{V}^T & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\]

Notice fewer matrix multiplies!
More simplifications

- **BUT**
  - $G_x$ is much simpler than it might look
  - each set of measurements affected by only one landmark!

\[
G_x = \begin{bmatrix}
\frac{\partial \theta_1}{\partial \mathbf{R}} & \frac{\partial \theta_1}{\partial \mathbf{L}_1} & 0 & 0 & 0 & 0 \\
\frac{\partial \theta_2}{\partial \mathbf{R}} & 0 & \frac{\partial \theta_2}{\partial \mathbf{L}_2} & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\frac{\partial \theta_N}{\partial \mathbf{R}} & 0 & 0 & 0 & 0 & \frac{\partial \theta_N}{\partial \mathbf{L}_N}
\end{bmatrix}
\]
More simplifications

• BUT
  • $G_n$ is usually much simpler than it might look
    • noise is usually additive normal noise

• This means that the term: $G_n \sum_{n,i} G_n^T$

• is actually a block diagonal matrix
Big simplification

• The nasty bit…

\[
\left[ g_x \Sigma_i - g_x^T + g_n \Sigma_{m,j} g_n^T \right]^{-1}
\]

• But notice key point
  • measurements interact only through the position/orientation of the vehicle
  • OR measurements are conditionally independent conditioned on pose of v.
  • OR you could subdivide time and update measurements one by one
  • OR matrix G_x has the sparsity structure above

• (the same point, manifesting in different ways)
Subdividing time…

• We receive measurements of landmarks in some order
  • a measurement of the position of landmark i affects the whole state
    • because it changes your estimate of the location of the vehicle
      • and that affects your estimate of state of every landmark
  • BUT
    • the change in estimate of location depends ONLY on
      • location
      • landmark i
Steps in EKF

\[ \mathcal{K}_i = \Sigma_i^{-} G_x^T \left[ G_x \Sigma_i^{-} G_x^T + G_n \Sigma_{m,i} G_n^T \right]^{-1} \]

\[ \mathbf{x}_i^+ = \mathbf{x}_i^- + \mathcal{K}_i \left[ \mathbf{y}_i - g(\mathbf{x}_i^-, 0) \right] \]

\[ \Sigma_i^+ = [\mathcal{I} - \mathcal{K}_i G_x] \Sigma_i^- \]
Steps in EKF

\[ \mathcal{K}_i = \sum_i -G_x^T \left[ G_x \sum_i -G_x^T + G_n \sum_{m,i} G_n^T \right]^{-1} \]

\[ \mathbf{x}_{i}^+ = \mathbf{x}_i^- + \mathcal{K}_i \left[ y_i - g(\mathbf{x}_i^-, 0) \right] \]

\[ \Sigma_i^+ = \left[ \mathcal{I} - \mathcal{K}_i G_x \right] \Sigma_i^- \]

One measurement from one landmark!

Notice:
Inverting only a small matrix

Notice:
But affecting the whole state!
Landmarks

• Which measurement comes from which landmark?
  • data association -
    • use some form of bipartite graph matching

• Idea:
  \[
  \overline{X}_i
  \]
  • predicts landmark positions, vehicle position before obs
  • compute distances between all pairs of
    • predicted obs, real obs
    • bipartite graph matcher
  • OR greedy
Landmarks

• No measurement from a landmark?
  • structure of EKF means you can process landmarks one by one
    • that’s what all the matrix surgery was about
    • so don’t update that landmark

• How do we know no measurement from a landmark?
  • refuse to match if distance in greedy/bipartite is too big
  • other kinds of matching problem (color, features, etc)
Measuring distance and orientation

- Landmark is at:
  - in global coordinate system
- We record distance and heading:
  - measurement

\[
\begin{bmatrix}
  d \\
  \phi
\end{bmatrix} = \begin{bmatrix}
  \sqrt{(x - u)^2 + (y - v)^2} \\
  \text{atan2}(y - u, x - v) - \theta
\end{bmatrix}
\]

THIS ISN’T LINEAR!
A further trick: inverting measurement

- Example: measure distance and orientation to point

\[ \begin{bmatrix} u \\ v \end{bmatrix} \]

point posn in world coords

\[ \begin{bmatrix} \sqrt{(x-u)^2 + (y-v)^2} \\ \text{atan2}(y-u, x-v) - \theta \end{bmatrix} \]

vehicle posn in world coords

vehicle orientation in world coords

Observation

\[ \begin{bmatrix} d \\ \phi \end{bmatrix} = \begin{bmatrix} \sqrt{(x-u)^2 + (y-v)^2} \\ \text{atan2}(y-u, x-v) - \theta \end{bmatrix} \]
Range and bearing

\[
\begin{bmatrix}
  d \\
  \phi
\end{bmatrix} = \begin{bmatrix}
  \sqrt{(x-u)^2 + (y-v)^2} \\
  \text{atan2}(y-u, x-v) - \theta
\end{bmatrix}
\]

Observation → 

\[
\begin{bmatrix}
  u \\
  v
\end{bmatrix} = \begin{bmatrix}
  x + (d + \xi) \sin(\phi + \zeta + \theta) \\
  y + (d + \xi) \cos(\phi + \zeta + \theta)
\end{bmatrix}
\]

These are measurements of landmark ONLY

Landmark position

Vehicle state

Noise affecting measurements

Here use the current estimate of vehicle state
Bearing only (sketch)

• Cannot determine landmark in 2D from measurement
  • it’s on a line!
  • you must come up with a prior
    • after that, it’s easy
      • find mean posterior location, covariance
      • plug in
  • Big Issue
    • True prior should have infinite covariance
      • can’t work with that
      • so linearization may fail