AUTONOMOUS SYSTEMS

Graph-based SLAM

Instituto Superior Técnico
MEEC · MEAer · MEIC
2016 / 2017

Rodrigo Ventura
(rodrigo.ventura@isr.ist.utl.pt)
Problem statement

- Consider a robot moving in an unknown environment, capable of perceiving static landmarks located in the environment
  - Robot poses: \( x_{1:t} \) denotes a sequence of poses \( x_1, ..., x_t \)
  - Landmarks: \( l_1, ..., l_N \) where \( N \) is unknown

- The full SLAM (Simultaneously Localization And Mapping) problem amounts to simultaneously estimating the poses \( x_{1:t} \) and the landmark locations \( l_{1:N} \), given the sequence of actions \( u_{1:t} \) and measurements \( z_{1:t} \)

- In a Bayesian framework this can be formulated as a maximization of a posterior conditional distribution

\[
(\widehat{x}_{1:t}, \widehat{l}_{1:N}) = \arg \max_{x_{1:t}, l_{1:N}} P(x_{1:t}, l_{1:N} | u_{1:t}, z_{1:t})
\]
Factorization of the posterior

- Stochastic model of system evolution and measurement

\[ x_t = g_t(x_{t-1}, u_t) + \varepsilon_t \]
\[ z_t = h_t(x_t, l_{c_t}) + \delta_t \]

where \( \varepsilon_t \sim N(0, Q_t) \), \( \delta_t \sim N(0, R_t) \), and \( c_t \) is the index of the landmark observed at time \( t \) (i.e., the correspondences)

- Since \( x_t \) only depends on \( x_{t-1} \) and \( u_t \), \( z_t \) only depends on \( x_t \), and the noise terms are uncorrelated, the posterior can be factored into

\[
P(x_{1:t}, l_{1:N} \mid u_{1:t}, z_{1:t}) \propto \prod_{i=1}^{t} P(z_i \mid x_i, l_{c_t}) \prod_{i=1}^{t} P(x_i \mid x_{i-1}, u_i)
\]
Graphical model view

- **Bayesian network**: Direct Acyclic Graph (DAG) where nodes are described by conditional probability distributions given the parent nodes.
Graphical model view

- **Markov network**: undirected graph where edges represent factors in the posterior factorization

![Graphical Model Diagram]

- $x_{i-1}$, $x_i$, $x_{i+1}$
- $u_i$, $u_{i+1}$
- $l_1$, $l_3$
- $z_{i-1}$, $z_{i}$, $z_{i+1}$
• Given that

\[ x_t = g_t(x_{t-1}, u_t) + \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, Q_t) \]

the distribution of \( x_t \) given \( x_{t-1} \) and \( u_t \) is normal \( \mathcal{N}(g_t(x_{t-1}, u_t), Q_t) \) with density

\[
P(x_t|x_{t-1}, u_t) = \eta \exp \left( \frac{-1}{2} \left( x_t - g_t(x_{t-1}, u_t) \right)^T Q_t^{-1} \left( x_t - g_t(x_{t-1}, u_t) \right) \right)
\]

\[ \eta = \frac{1}{\sqrt{(2\pi)^k |Q_t|}} \]
Measurement model

- Given that

\[ z_t = h_t(x_t, l_{ct}) + \delta_t \]

the distribution of \( z_t \) given \( x_t \) and landmark \( l_{ct} \) is normal \( N(h_t(x_t, l_{ct}), R_t) \) with density

\[
P(z_t|x_t, l_{ct}) = \eta \exp \left( -\frac{1}{2} (z_t - h_t(x_t, l_{ct}))^T R_t^{-1} (z_t - h_t(x_t, l_{ct})) \right)
\]

\[
\eta = \frac{1}{\sqrt{(2\pi)^k |R_t|}}
\]
Optimization problem

- The solution to the full SLAM problem amounts to a maximization problem that can be algebraically simplified into

\[
\left( \hat{x}_{1:t}, \hat{l}_{1:N} \right) = \arg \max_{x_{1:t}, l_{1:N}} P(x_{1:t}, l_{1:N} | u_{1:t}, z_{1:t})
\]

\[
= \arg \min_{x_{1:t}, l_{1:N}} - \log \prod_{i=1}^{t} P(z_i | x_i, l_{c_t}) \prod_{i=1}^{t} P(x_i | x_{i-1}, u_i)
\]

\[
= \arg \min_{x_{1:t}, l_{1:N}} \sum_{i=1}^{t} (x_t - g_t(x_{t-1}, u_t))^T Q_t^{-1} (x_t - g_t(x_{t-1}, u_t)) + \\
\sum_{i=1}^{t} (z_t - h_t(x_t, l_{c_t}))^T R_t^{-1} (z_t - h_t(x_t, l_{c_t}))
\]

this is an hard nonlinear optimization problem
Linearization

- Linearizing the motion model function $g()$ at the initial estimate $x^0_{t-1}$

$$g_t(x_{t-1}, u_t) \approx g_t(x^0_{t-1}, u_t) + G_t (x_{t-1} - x^0_{t-1})$$

$$= G_t x_{t-1} - a_t$$

where $a_t = x^0_{t-1} - g_t(x^0_{t-1}, u_t)$ and $G_t$ is the Jacobian matrix

$$G_t = \left. \frac{\partial g_t(x, u_t)}{\partial x} \right|_{x=x^0_{t-1}}$$

- Then, the corresponding quadratic term becomes

$$(x_t - g_t(x_{t-1}, u_t))^T Q_t^{-1} (x_t - g_t(x_{t-1}, u_t))$$

$$\approx (x_t - G_t x_{t-1} - a_t)^T Q_t^{-1} (x_t - G_t x_{t-1} - a_t)$$
Linearization

- Linearizing the measurement model function \( h() \) both at the initial estimates of \( x_0^t \) and \( l_0^ct \)

\[
h_t(x_t, l_{ct}) \simeq h_t(x_0^t, l_0^ct) + H_t (x_t - x_0^t) + J_t (l_{ct} - l_0^ct)
\]

\[
= H_t x_t + J_t l_{ct} - H_t x_0^t - J_t l_0^ct + h_t(x_0^t, l_0^ct)
\]

where \( H_t \) and \( J_t \) are the Jacobian matrices

\[
H_t = \left. \frac{\partial h_t(x, l_{ct})}{\partial x} \right|_{x=x_0^t} \quad J_t = \left. \frac{\partial h_t(x_0^t, l)}{\partial l} \right|_{l=l_0^ct}
\]

- Then, the corresponding quadratic term becomes

\[
(z_t - h_t(x_t, l_{ct}))^T R_t^{-1} (z_t - h_t(x_t, l_{ct}))
\]

\[
\simeq (H_t x_t + J_t l_{ct} - b_t)^T R_t^{-1} (H_t x_t + J_t l_{ct} - b_t)
\]

where \( b_t = H_t x_0^t + J_t l_0^ct - h_t(x_0^t, l_0^ct) - z_t \)
Putting it all together

- After linearization every quadratic term is in the form

\[
(A_k y_k - e_k)^T \Omega_k (A_k y_k - e_k), \quad k = 1, \ldots
\]

where \(A_k\) is a block matrix, \(y_k\) is a vector containing a subset of the variables, \(\Omega_k\) is an inverse of a covariance (i.e., an information matrix), and \(e_k\) is a vector.

- Therefore:

\[
(x_t - G_t x_{t-1} - a_t)^T Q_t^{-1} (x_t - G_t x_{t-1} - a_t)
\]

\[
= \left( \begin{bmatrix} I & 0 \\ 0 & -G_t \end{bmatrix} \right) \left( \begin{bmatrix} x_t \\ x_{t-1} \end{bmatrix} - a_t \right)^T Q_t^{-1} \left( \begin{bmatrix} I & 0 \\ 0 & -G_t \end{bmatrix} \right) \left( \begin{bmatrix} x_t \\ x_{t-1} \end{bmatrix} - a_t \right)
\]

\[
(H_t x_t + J_t l_{ct} - b_t)^T R_t^{-1} (H_t x_t + J_t l_{ct} - b_t)
\]

\[
= \left( \begin{bmatrix} H_t & 0 \\ 0 & J_t \end{bmatrix} \right) \left( \begin{bmatrix} x_t \\ l_{ct} \end{bmatrix} - b_t \right)^T R_t^{-1} \left( \begin{bmatrix} H_t & 0 \\ 0 & J_t \end{bmatrix} \right) \left( \begin{bmatrix} x_t \\ l_{ct} \end{bmatrix} - b_t \right)
\]
Recalling the graphical view

- Generic quadratic form: \((A_k y_k - e_k)^T \Omega_k (A_k y_k - e_k)\)

  for motion model factor

  \[
  A_k = \begin{bmatrix} I & 0 \\ 0 & -G_t \end{bmatrix}, \quad y_k = \begin{pmatrix} x_t \\ x_{t-1} \end{pmatrix}, \quad e_k = a_t, \quad \Omega_k = Q_t^{-1}
  \]

  for the measurement model factor

  \[
  A_k = \begin{bmatrix} H_t & 0 \\ 0 & J_t \end{bmatrix}, \quad y_k = \begin{pmatrix} x_t \\ l_{ct} \end{pmatrix}, \quad e_k = b_t, \quad \Omega_k = R_t^{-1}
  \]
Putting it all together

- Let $\theta$ be a vector with all the variables to estimate: 
  \[ \theta = \begin{pmatrix} x_1 \\ \vdots \\ x_t \\ l_1 \\ \vdots \\ l_N \end{pmatrix} \]

- Since all $y_k$ vectors are a subset of $\theta$, each quadratic term can be put as 
  \[ (B_k \theta - e_k)^T \Omega_k (B_k \theta - e_k), \quad k = 1, \ldots \]
  where $B_k$ is a matrix with the rows of $A_k$ such that $B_k \theta = A_k y_k$

- Therefore, the approximate solution becomes 
  \[ \left( \hat{x}_{1:t}, \hat{l}_{1:N} \right) \approx \arg \min_{x_{1:t}, l_{1:N}} \sum_{k} (B_k \theta - e_k)^T \Omega_k (B_k \theta - e_k) \]
Solving the optimization problem

- There are two main methods for solving the non-linear optimization problem
  - **GraphSLAM**, proposed by Sebastian Thrun et al in 2006
    
  
  - **iSAM**, proposed by Michael Kaess et al in 2008
    

- Both approaches are iterative, comprising two steps:
  1. obtaining an approximate closed-form solution of the linearized problem
  2. re-linearization at the improved estimates obtained in step (1)
GraphSLAM

- GraphSLAM is based on observing that the cost function to be minimized

\[
\sum_k (B_k \theta - e_k)^T \Omega_k (B_k \theta - e_k) = \sum_k \left( \theta^T B_k^T \Omega_k B_k \theta - 2 e_k^T \Omega_k B_k \theta + e_k^T e_k \right)
\]

\[
= \theta^T \left( \sum_k B_k^T \Omega_k B_k \right) \theta - 2 \left( \sum_k e_k^T \Omega_k B_k \right) \theta + \sum_k e_k^T e_k
\]

\[
= \theta^T A \theta - 2b^T \theta + c
\]

is a simple quadratic form with a closed form solution

\[
\frac{\partial}{\partial \theta} \left( \theta^T A \theta - 2b^T \theta + c \right) = 2A \theta - 2b = 0
\]

\[
\hat{\theta} = A^{-1} b = \left( \sum_k B_k^T \Omega_k B_k \right)^{-1} \left( \sum_k e_k^T \Omega_k B_k \right)
\]
GraphSLAM

- Each edge $k$ in the Markov network maps to a pair of terms $B_k^T \Omega_k B_k$ and $e_k^T \Omega_k B_k$ in the solution

$$\hat{\theta} = \left( \sum_k B_k^T \Omega_k B_k \right)^{-1} \left( \sum_k e_k^T \Omega_k B_k \right)$$

- Example: total 5 factors = 2 from motion + 3 from measurement
GraphSLAM

• GraphSLAM is extremely flexible, since it can be directly applied to any Markov network. Examples:
  
  – many measurements at a single pose
  
  – only pose constraints (e.g., scan matching)
  
  – multiple robots
  
  – moving targets
  
  – etc.
iSAM

- iSAM stands for Incremental Smoothing and Mapping
- iSAM views the cost function as a Least-Squares (LS) problem

\[
\sum_{k} (B_k \theta - e_k)^T \Omega_k (B_k \theta - e_k) = \sum_{k} \left[ \Omega_k^{1/2} (B_k \theta - e_k) \right]^T \left[ \Omega_k^{1/2} (B_k \theta - e_k) \right] \\
= \sum_{k} \left\| \Omega_k^{1/2} B_k \theta - \Omega_k^{1/2} e_k \right\|^2 \\
= \left\| \begin{bmatrix} \Omega_1^{1/2} B_1 \\ \vdots \\ \Omega_M^{1/2} B_M \end{bmatrix} \theta - \begin{bmatrix} \Omega_1^{1/2} e_1 \\ \vdots \\ \Omega_M^{1/2} e_M \end{bmatrix} \right\|^2 \\
= \| A \theta - b \|^2
\]

- Note that \( A \) and \( b \) can be built incrementally as new measurements arrive
iSAM

- Least-Squares problem:

  \[ \hat{\theta} = \arg \min_{\theta} \| A \theta - b \|^2 \]

- Since \( A \) has more rows than columns, which makes \( A\theta = b \) a over-specified linear system

- QR factorization: any matrix \( A \) can be decomposed as

  \[ A = Q \begin{bmatrix} R \\ 0 \end{bmatrix} \]

  where \( Q \) is an orthogonal matrix and \( R \) is an upper triangular matrix

- Thus:

  \[ \| A \theta - b \|^2 = \| R \theta - d \|^2 + \| e \|^2 \]

  \[ \arg \min_{\theta} \| A \theta - b \|^2 = \arg \min_{\theta} \| R \theta - d \|^2 \]

  \[ R \hat{\theta} = d \]

  which can be trivially solved using simple back-substitution
QR factorization can be done incrementally:

- the Givens transformation transforms non-upper-triangular matrices into upper-triangular ones by successively left-multiplying them with orthogonal matrices based on

\[
\Phi = \begin{bmatrix}
\cos \phi & \sin \phi \\
-\sin \phi & \cos \phi 
\end{bmatrix}
\]

- illustration:

\[
\begin{array}{c}
\text{row } k \\
\text{row } i
\end{array}
\begin{array}{c}
\begin{array}{ccc}
\cdots & 1 & \text{c} \\
\text{s} & \cdots & 1 \\
\text{-s} & \text{c} & \cdots
\end{array}
\end{array}
\begin{array}{c}
\text{Givens} \\
\text{R} \\
\text{R'}
\end{array}
\]

\[
= \begin{array}{c}
x \\
\end{array}
\]

\[
= \begin{array}{c}
\text{0} \\
\end{array}
\]

\[
\text{Autonomous Systems  —  course slides  —  Rodrigo Ventura}
\]
new rows added

Givens transformation

iSAM

new rows at t=1

R

rhs

+  

new rows at t=2

+  

t=2

+  

t=3

+  

t=50

+  

Examples of such updates are shown in Fig. 3. The simulation for a linear exploration task in iSAM allows relinearization of the trajectory, as well as with nonlinear sensor measurement equations, and suggests a solution in connection with periodic variable reordering. We show how iSAM allows relinearization of robot heading. We show how iSAM allows relinearization of robot heading.
iSAM

- Matrix $R$ can be made sparse by variable re-ordering

- This is particularly important after loop closure

(b) Factor $R$.  
(c) The same factor $R$ after variable reordering.
• Results:

Fig. 8. Results for the full Victoria Park sequence. Solving the complete problem including data association in each step took 7.7 minutes on a laptop computer. For known correspondences, the time reduces to 5.9 minutes. Since the dataset is from a 26 minute long robot run, iSAM with unknown correspondences is over 3 times faster than real-time in this case, calculating the complete and exact solution in each step. The trajectory and landmarks are shown in yellow (light), manually overlaid on an aerial image for reference. Differential GPS was not used in obtaining our experimental results, but is shown in blue (dark) for comparison - note that in many places it is not available.
iSAM

- More results:

(a) Trajectory based on odometry only.  
(b) Final trajectory and evidence grid map.  
(c) Final R factor with side length 2730.

Fig. 10. Results from iSAM applied to the Intel dataset. iSAM calculates the full solution for 910 poses and 4453 constraints with an average of 85\,ms per step, while reordering the variables every 20 steps. The problem has 910 \times 3 = 2730 variables and 4453 \times 3 = 13,359 measurement equations. The \( R \) factor contains 90,363 entries, which corresponds to 2.42\% or 33.1 entries per column.

(a) Trajectory based on odometry only.  
(b) Final trajectory and evidence grid map.  
(c) Final R factor with side length 5823.

Fig. 11. iSAM results for the MIT Killian Court dataset. iSAM calculates the full solution for the 1941 poses and 2190 pose constraints with an average of 12.2\,ms per step. The \( R \) factor contains 52,414 entries for 5823 variables, which corresponds to 0.31\% or 9.0 per column.