Probabilistic Robotics: Localization & SLAM

METR 4202: **Robotics** & Automation

Dr Surya Singh -- Lecture # 12

October 18, 2017

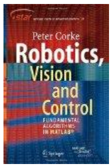
metr4202-staff@itee.uq.edu.au

http://robotics.itee.uq.edu.au/~metr4202/ [http://**metr4202.com**]

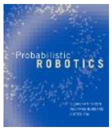
Lecture Schedule

Week	Date	Lecture (W: 3:05p-4:50, 7-222)
1	26-Jul	Introduction + Representing Position & Orientation & State
2	2-Aug	Robot Forward Kinematics (Frames, Transformation Matrices & Affine Transformations)
3	9-Aug	Robot Inverse Kinematics & Dynamics (Jacobians)
4	16-Aug	*Ekka Day* (Robot Kinematics & Kinetics Review)
5	23-Aug	Jacobians & Robot Sensing Overview
6	30-Aug	Robot Sensing: Single View Geometry & Lines
7	6-Sep	Robot Sensing: Basic Feature Detection
8	13-Sep	Robot Sensing: Scalable Feature Detection
9	20-Sep	Mid-Semester Exam & Multiple View Geometry
	27-Sep	*Study break*
10	4-Oct	Motion Planning
11	11-Oct	Probabilistic Robotics: Planning & Control (Sample-Based Planning/State-Space/LQR)
12	18-Oct	**Probabilistic Robotics: Localization & SLAM**
13	25-Oct	The Future of Robotics/Automation + Challenges + Course Review

Follow Along Reading:



Robotics, Vision & Control
by Peter Corke
Also online: *SpringerLink*
UQ Library eBook: 364220144X



Probabilistic robotics
by Thrun, Burgard, and Fox

UQ Library: TJ211 .T575 2005

Today

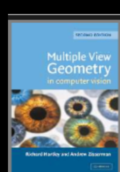
➔ SLAM 🡐

- RVC:
  - pp. 123-4 (§6.4-6.5)
- Probabilistic robotics
  - pp. 309-382 (§10.1-11.11)

- Everything ☺
  - (It's a review/recap lecture)

Next Time

Structure from Motion!

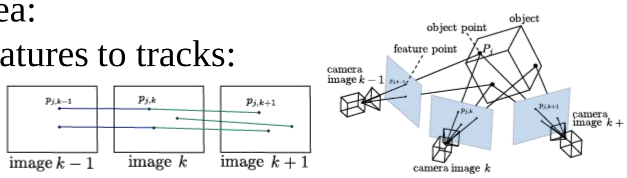


SFM: Structure from Motion

Source: C. Ham, *Handheld Monocular Object Reconstruction – Uniting Photogrammetry, Silhouettes, and Scale*, PhD Thesis 2017

Motivating SFM: OpenMVG
http://imagine.enpc.fr/~moulonp/openMVG/

- Core Idea:
  From features to tracks:

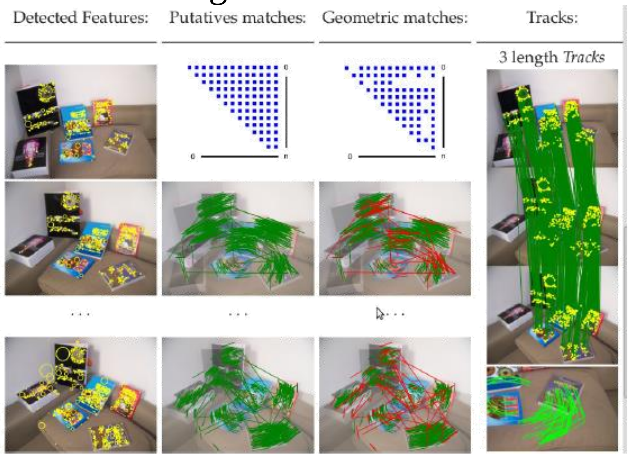


Source: OpenMVG Docs [http://openmvg.readthedocs.io/en/latest/openMVG/tracks/tracks/]

Motivating SFM: OpenMVG

http://imagine.enpc.fr/~moulonp/openMVG/

- Feature based tracking:



Source: OpenMVG Docs [http://openmvg.readthedocs.io/en/latest/openMVG/tracks/tracks/]

Structure [from] Motion

- Given a set of feature tracks,
  estimate the 3D structure and 3D (camera) motion.

- Assumption: orthographic projection

- Tracks: (u_{fp}, v_{fp}), f: frame, p: point
- Subtract out **mean** 2D position…

  $\mathbf{i}_f$: rotation, $\mathbf{s}_p$: position

$$u_{fp} = i_f^T s_p, v_{fp} = j_f^T s_p$$

From Szeliski, *Computer Vision: Algorithms and Applications*

Structure from motion

- How many points do we need to match?
- 2 frames:
  - (R,t): 5 dof + 3n point locations ≤
  - 4n point measurements ⇒
  - n ≥ 5

$$\hat{u}_{ij} = f(\mathrm{K}, \mathrm{R}_j, \mathrm{t}_j, \mathrm{x}_i)$$
$$\hat{v}_{ij} = g(\mathrm{K}, \mathrm{R}_j, \mathrm{t}_j, \mathrm{x}_i)$$

- k frames:
  - 6(k–1)-1 + 3n ≤ 2kn
- always want to use many more

From Szeliski, *Computer Vision: Algorithms and Applications*

Measurement equations

- Measurement equations

$$u_{fp} = \boldsymbol{i}_f^T \boldsymbol{s}_p \qquad \boldsymbol{i}_f\text{: rotation, } \boldsymbol{s}_p\text{: position}$$
$$v_{fp} = \boldsymbol{j}_f^T \boldsymbol{s}_p$$

- Stack them up…

$$\boldsymbol{W} = \boldsymbol{R}\ \boldsymbol{S}$$
$$\boldsymbol{R} = (\boldsymbol{i}_1,\ldots,\boldsymbol{i}_F, \boldsymbol{j}_1,\ldots,\boldsymbol{j}_F)^T$$
$$\boldsymbol{S} = (\boldsymbol{s}_1,\ldots,\boldsymbol{s}_P)$$

From Szeliski, *Computer Vision: Algorithms and Applications*

Factorization

$$\boldsymbol{W} = \boldsymbol{R}_{2F\times 3}\,\boldsymbol{S}_{3\times P}$$

SVD

$$\boldsymbol{W} = \boldsymbol{U}\,\boldsymbol{\Lambda}\,\boldsymbol{V} \quad \boldsymbol{\Lambda} \textbf{ must be rank 3}$$

$$\boldsymbol{W'} = (\boldsymbol{U}\boldsymbol{\Lambda}^{1/2})(\boldsymbol{\Lambda}^{1/2}\,\boldsymbol{V}) \quad = \boldsymbol{U'}\,\boldsymbol{V'}$$

Make ***R*** orthogonal

$$\boldsymbol{R} = \boldsymbol{Q}\boldsymbol{U'},\ \ \boldsymbol{S} = \boldsymbol{Q}^{-1}\boldsymbol{V'}$$

$$\boldsymbol{i}_f^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{i}_f = 1\ \ldots$$

From Szeliski, *Computer Vision: Algorithms and Applications*

Results

- Look at paper figures…

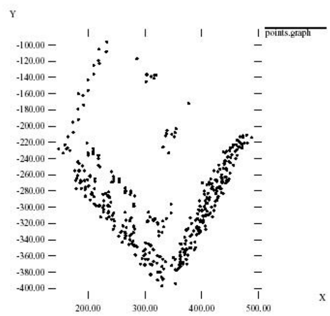


Figure 4.5: A view of the computed shape from approximately above the building (compare with figure 4.6).

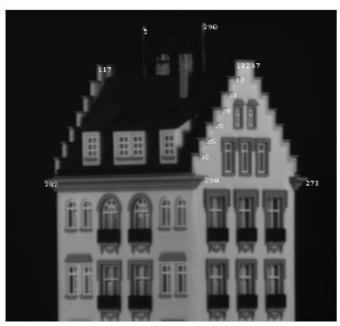


Figure 4.7: For a quantitative evaluation, distances between the features shown in the picture were measured on the actual model, and compared with the computed results. The comparison is shown in figure 4.8.

From Szeliski, *Computer Vision: Algorithms and Applications*

Bundle Adjustment

- What makes this non-linear minimization hard?
  - many more parameters: potentially slow
  - poorer conditioning (high correlation)
  - potentially lots of outliers
  - gauge (coordinate) freedom

$$\begin{aligned} \hat{u}_{ij} &= f(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i) \\ \hat{v}_{ij} &= g(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i) \end{aligned}$$

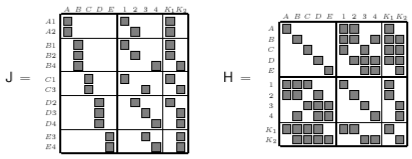
From Szeliski, *Computer Vision: Algorithms and Applications*

Lots of parameters: sparsity

- Only a few entries in Jacobian are non-zero

$$\begin{aligned} \hat{u}_{ij} &= f(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i) \\ \hat{v}_{ij} &= g(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i) \end{aligned}$$

$$\frac{\partial \hat{u}_{ij}}{\partial \mathbf{K}}, \quad \frac{\partial \hat{u}_{ij}}{\partial \mathbf{R}_j}, \quad \frac{\partial \hat{u}_{ij}}{\partial \mathbf{t}_j}, \quad \frac{\partial \hat{u}_{ij}}{\partial \mathbf{x}_i},$$

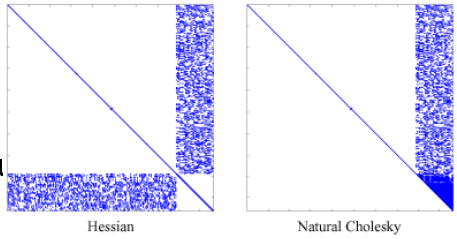


From Szeliski, *Computer Vision: Algorithms and Applications*

Sparse Cholesky (skyline)

- First used in finite element analysis
- Applied to SfM by [Szeliski & Kang 1994]

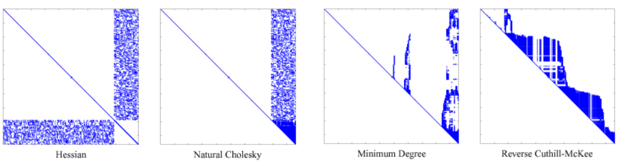
structu



From Szeliski, *Computer Vision: Algorithms and Applications*

Conditioning and gauge freedom

- Poor conditioning:
  - use 2nd order method
  - use Cholesky decomposition

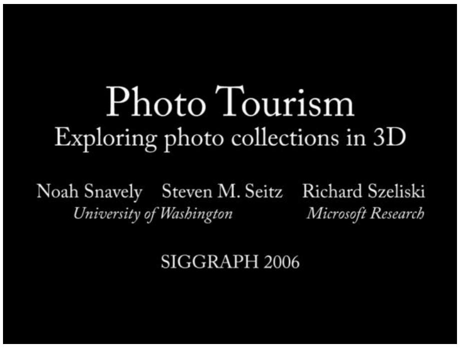


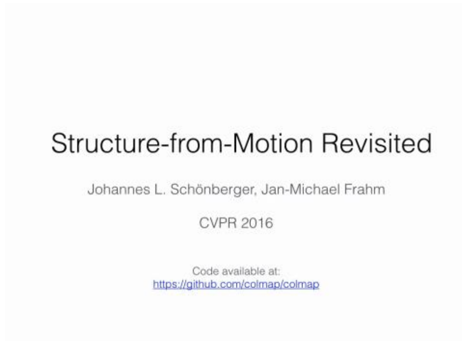
- Gauge freedom
  - fix certain parameters (orientation) or
  - zero out last few rows in Cholesky decomposition

From Szeliski, *Computer Vision: Algorithms and Applications*

More SFM: Cool Robotics Share!

- PhotoTourism
- [COLMAP](https://github.com/colmap/colmap)





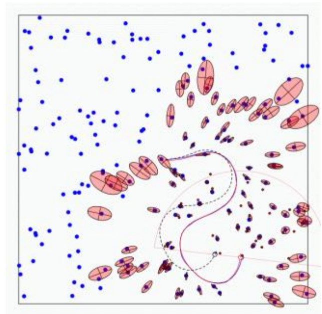
SLAM!

(Better than SMAL! ☺)

The SLAM Problem

A (robot) is exploring an **unknown**, **static** environment

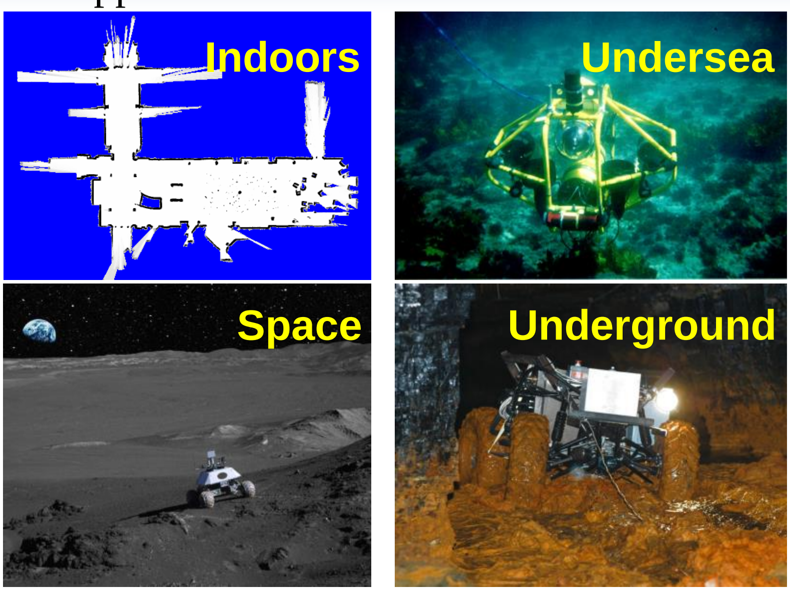
- Given:
  - The robot's controls
  - Observations of nearby features



- Problem:
  To Estimate:
  - The Location (*map*) of features
  - The Motion (*path*) of the robot

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

SLAM Applications



Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

What is SLAM?

- SLAM asks the following question:
  - Is it possible for an autonomous vehicle to start at an unknown location in an unknown environment and then to incrementally build a map of this environment while simultaneously using this map to compute vehicle location?

- SLAM has many indoor, outdoor, in-air and underwater applications for both manned and autonomous vehicles.

- Examples
  - Explore and return to starting point
  - Learn trained paths to different goal locations
  - Traverse a region with complete coverage (e.g., mine fields, lawns, reef monitoring)

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

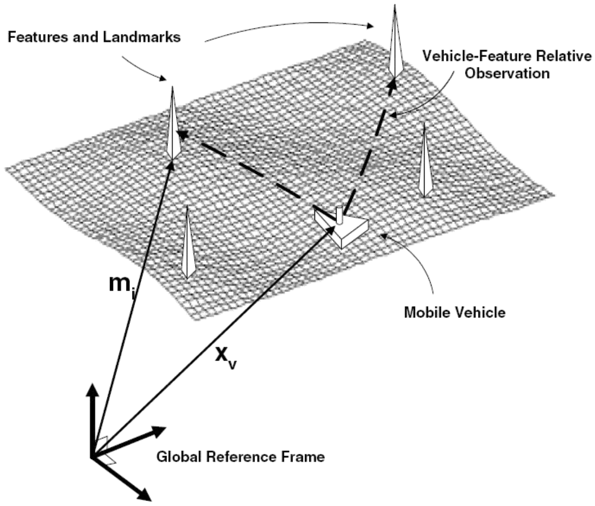
Components of SLAM

- Localisation
  - Determine pose given a priori map
- Mapping
  - Generate map when pose is accurately known from auxiliary source.

- SLAM
  - Define some arbitrary coordinate origin
  - Generate a map from on-board sensors
  - Compute pose from this map
  - Errors in map and in pose estimate are dependent.

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

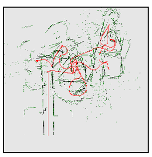
Structure of the Landmark-based SLAM-Problem

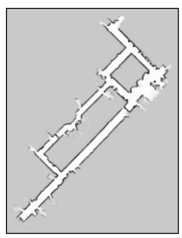


Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

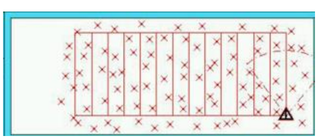
Representations

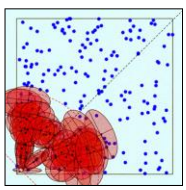
- Grid maps or scans

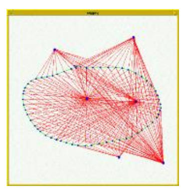




- Landmark-based





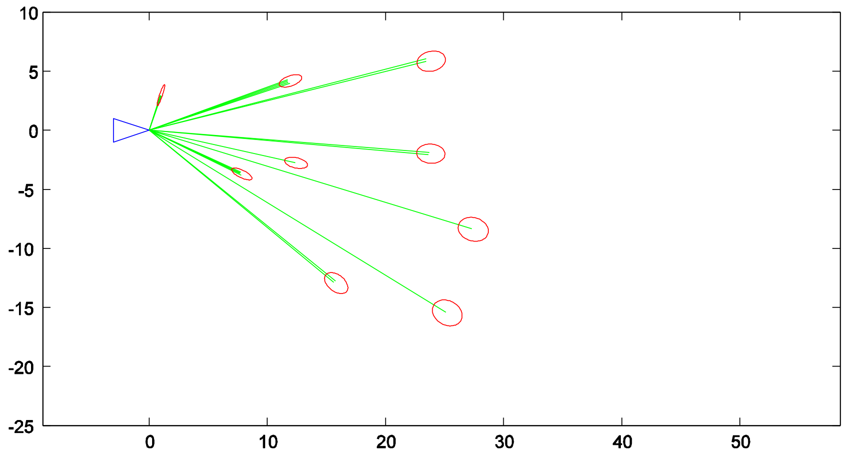


References: Grid Maps: [Lu & Milios, 97; Gutmann, 98: Thrun 98; Burgard, 99; Konolige & Gutmann, 00; Thrun, 00; Arras, 99; Haehnel, 01;…]
Landmark: [Leonard *et al.*, 98; Castelanos *et al.*, 99: Dissanayake *et al.*, 2001; Montemerlo *et al.*, 2002;…

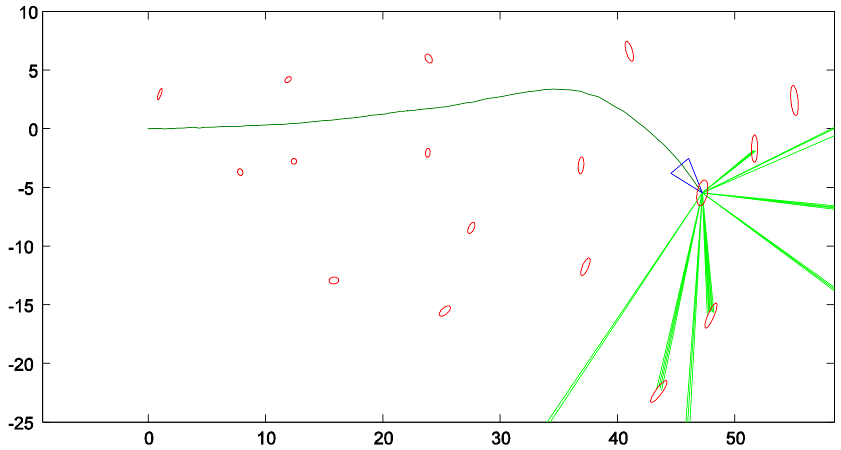
Basic SLAM Operation

Example: SLAM in Victoria Park

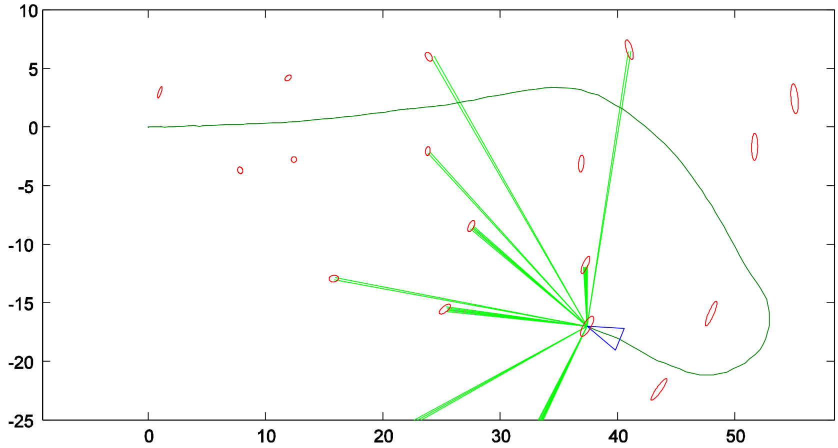
Basic SLAM Operation



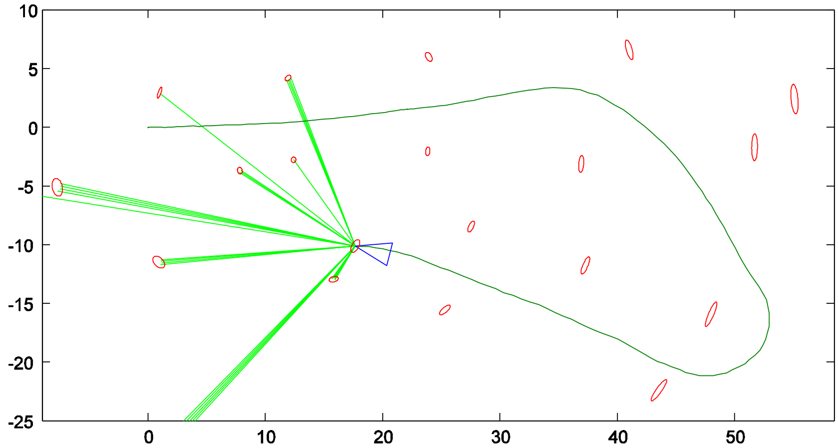
Basic SLAM Operation



Basic SLAM Operation

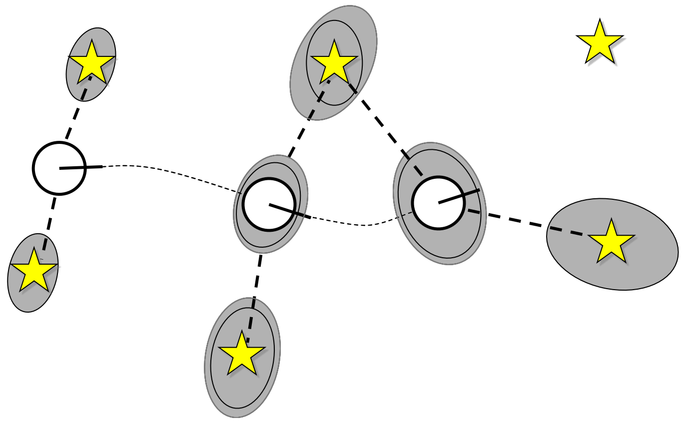


Basic SLAM Operation



Why is SLAM a hard problem?

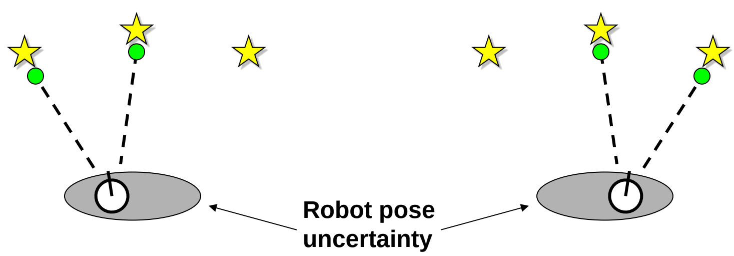
SLAM: robot path and map are both **unknown**



➔ Robot path error correlates errors in the map

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

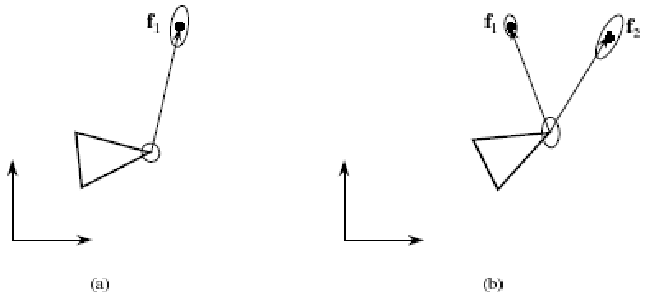
Why is SLAM a hard problem?



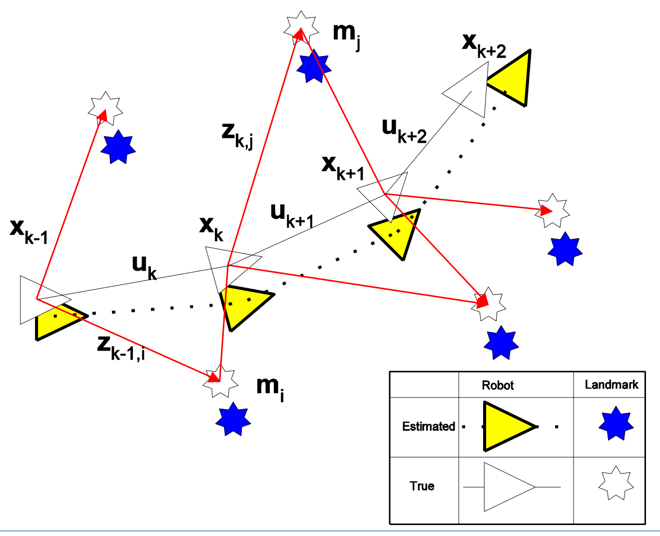
- In general, the mapping between observations and landmarks is unknown
- Picking wrong data associations can have catastrophic consequences
- **Pose error correlates data associations**

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

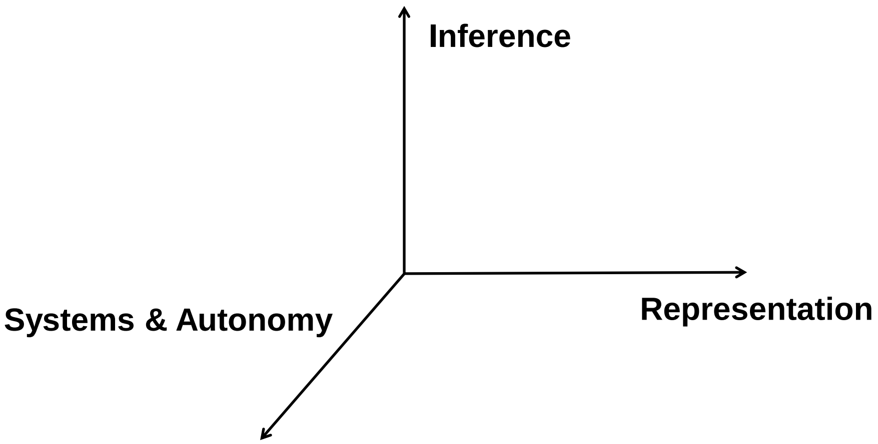
Dependent Errors



Correlated Estimates

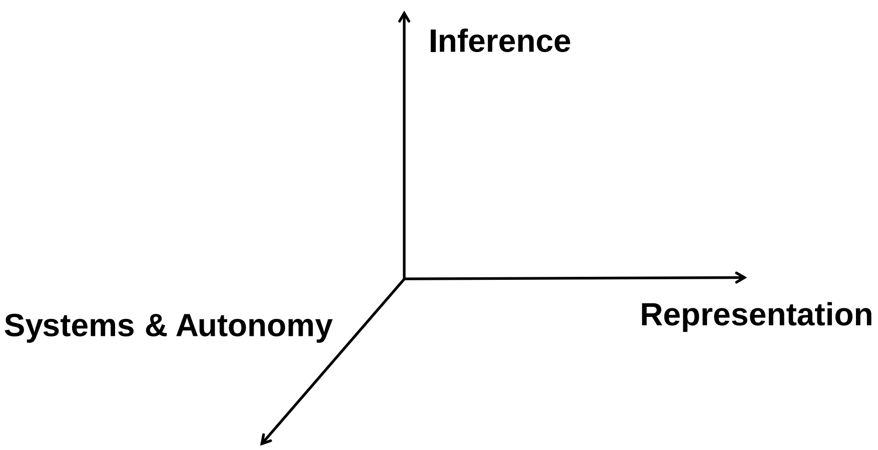


Why is SLAM Difficult?



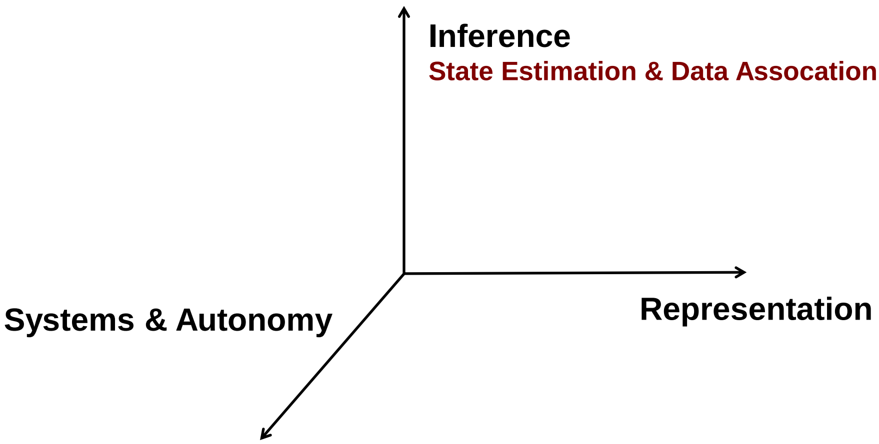
Source: Leonard (MIT)

Why is SLAM Difficult?



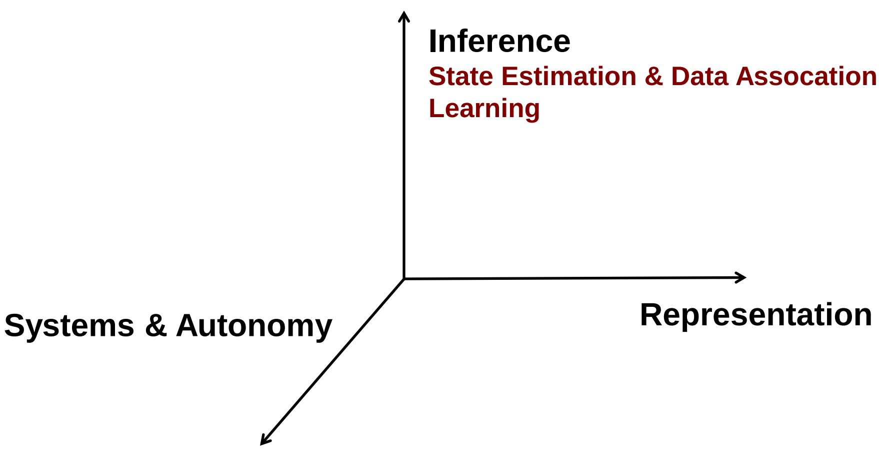
Source: Leonard (MIT)

Why is SLAM Difficult?



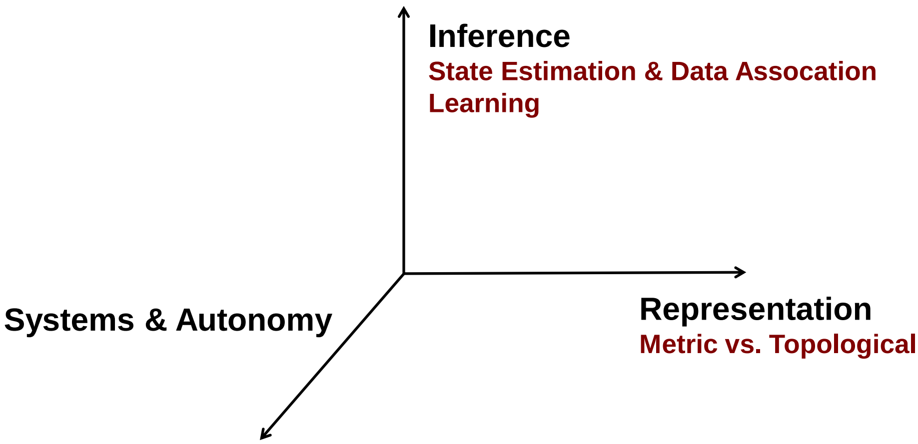
Source: Leonard (MIT)

Why is SLAM Difficult?



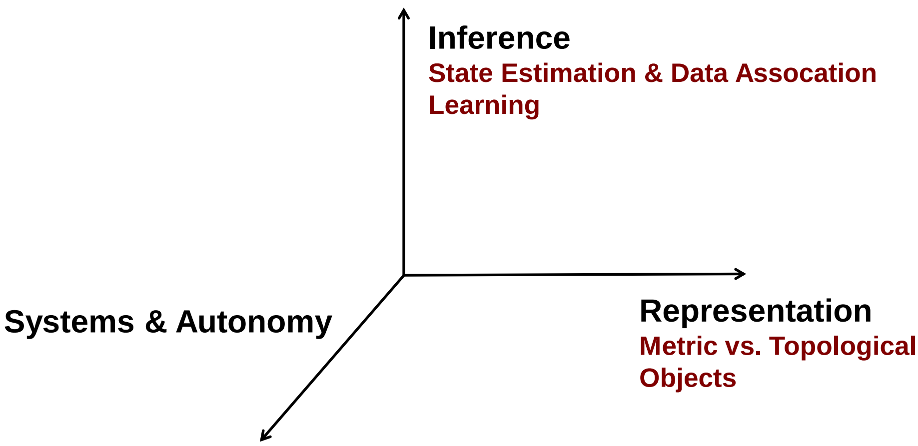
Source: Leonard (MIT)

Why is SLAM Difficult?



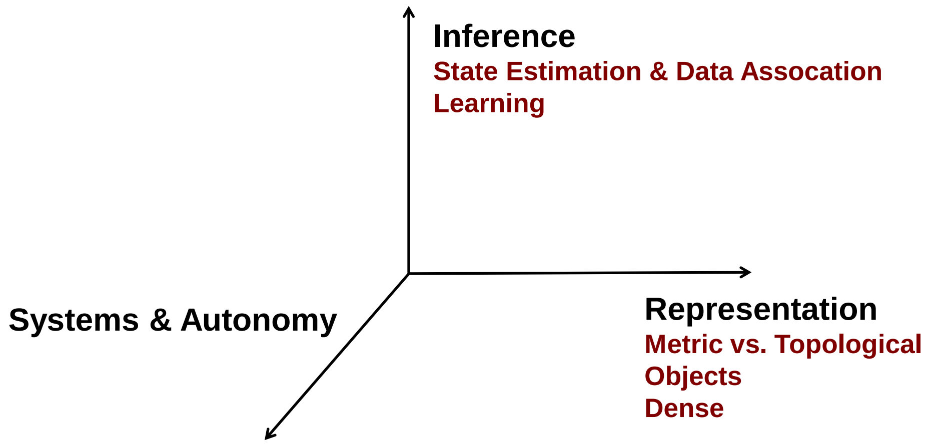
Source: Leonard (MIT)

Why is SLAM Difficult?



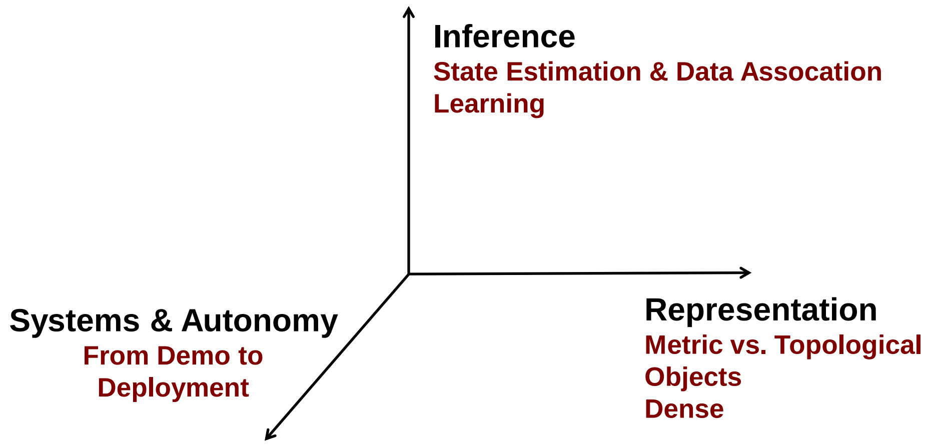
Source: Leonard (MIT)

Why is SLAM Difficult?



Source: Leonard (MIT)

Why is SLAM Difficult?



Source: Leonard (MIT)

SLAM: Simultaneous Localization and Mapping

- Online SLAM:

$$p(x_t, m \mid z_{1:t}, u_{1:t}) = \int\int\ldots\int p(x_{1:t}, m \mid z_{1:t}, u_{1:t})\, dx_1 dx_2 \ldots dx_{t-1}$$

  - Integrations typically done one at a time
  - Estimates most recent pose and map

- (*cf.*) "Full SLAM"

$$p(x_{1:t}, m \mid z_{1:t}, u_{1:t})$$

➔Estimates entire path and map (batch, like SFM)!

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

Scan Matching

- Maximize the likelihood of the i-th pose and map relative to the $(\boldsymbol{i}-\mathbf{1})$-th pose and map.

$$\hat{x}_t = \arg\max_{x_t} \left\{ p(z_t \mid x_t, \hat{m}^{[t-1]}) \cdot p(x_t \mid u_{t-1}, \hat{x}_{t-1}) \right\}$$

current measurement

robot motion

map constructed so far

- Calculate the map $\hat{m}^{[t]}$ according to "mapping with known poses" based on the poses and observations.

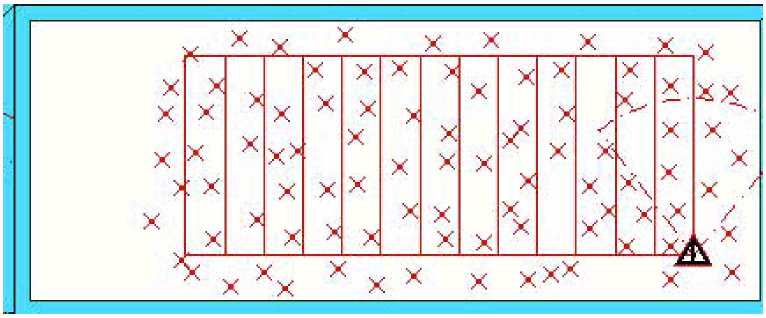
Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

Approximations for SLAM

- Local submaps
  [Leonard et al.99, Bosse et al. 02, Newman et al. 03]
- Sparse links (correlations)
  [Lu & Milios 97, Guivant & Nebot 01]
- Sparse extended information filters
  [Frese et al. 01, Thrun et al. 02]
- Thin junction tree filters
  [Paskin 03]
- Rao-Blackwellisation (FastSLAM)
  [Murphy 99, Montemerlo et al. 02, Eliazar et al. 03, Haehnel et al. 03]

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

Sub-maps for EKF SLAM



Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides
Reference: [Leonard, *et al.* 1998]

(E)KF-SLAM

- Map with ***N*** landmarks:
  $(3 + 2N)$-dimensional Gaussian

$$Bel(x_t, m_t) = \left\langle \begin{pmatrix} x \\ y \\ \theta \\ l_1 \\ l_2 \\ \vdots \\ l_N \end{pmatrix}, \begin{pmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\theta} & \sigma_{xl_1} & \sigma_{xl_2} & \cdots & \sigma_{xl_N} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{y\theta} & \sigma_{yl_1} & \sigma_{yl_2} & \cdots & \sigma_{yl_N} \\ \sigma_{x\theta} & \sigma_{y\theta} & \sigma_\theta^2 & \sigma_{\theta l_1} & \sigma_{\theta l_2} & \cdots & \sigma_{\theta l_N} \\ \sigma_{xl_1} & \sigma_{yl_1} & \sigma_{\theta l_1} & \sigma_{l_1}^2 & \sigma_{l_1 l_2} & \cdots & \sigma_{l_1 l_N} \\ \sigma_{xl_2} & \sigma_{yl_2} & \sigma_{\theta l_2} & \sigma_{l_1 l_2} & \sigma_{l_2}^2 & \cdots & \sigma_{l_2 l_N} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{xl_N} & \sigma_{yl_N} & \sigma_{\theta l_N} & \sigma_{l_1 l_N} & \sigma_{l_2 l_N} & \cdots & \sigma_{l_N}^2 \end{pmatrix} \right\rangle$$

- Can handle hundreds of dimensions

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

(Extended) Kalman Filter Algorithm

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

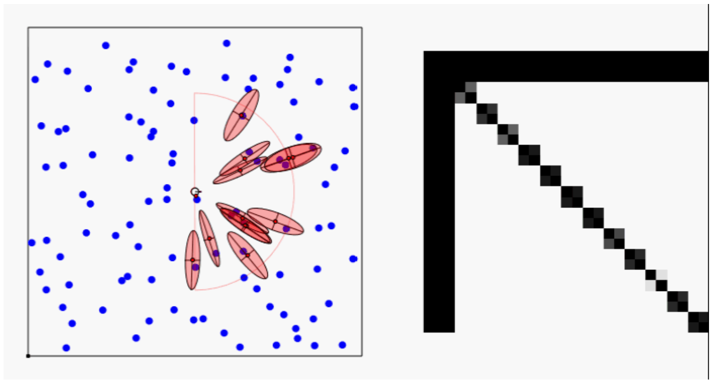
2. Prediction:
3. $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$
4. $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$

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

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

Classical Solution – The EKF

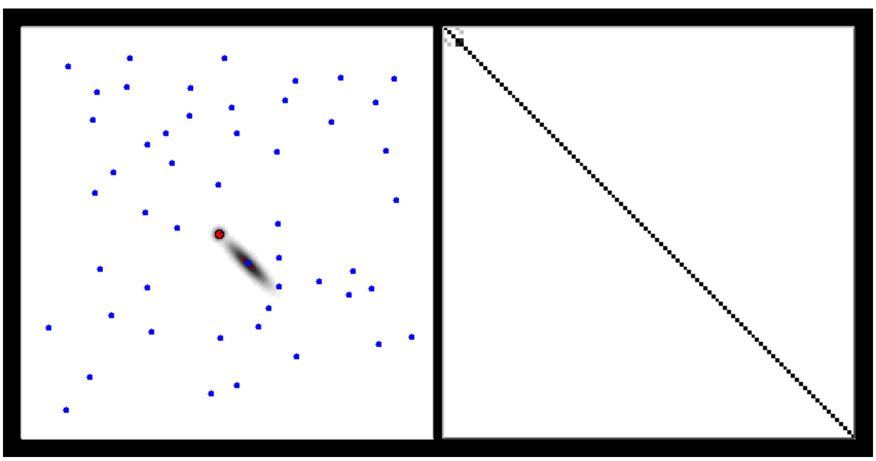


Blue path = true path   **Red path** = estimated path   **Black path** = odometry

- Approximate the SLAM posterior with a high-dimensional Gaussian [Smith & Cheesman, 1986] …
- Single hypothesis data association

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

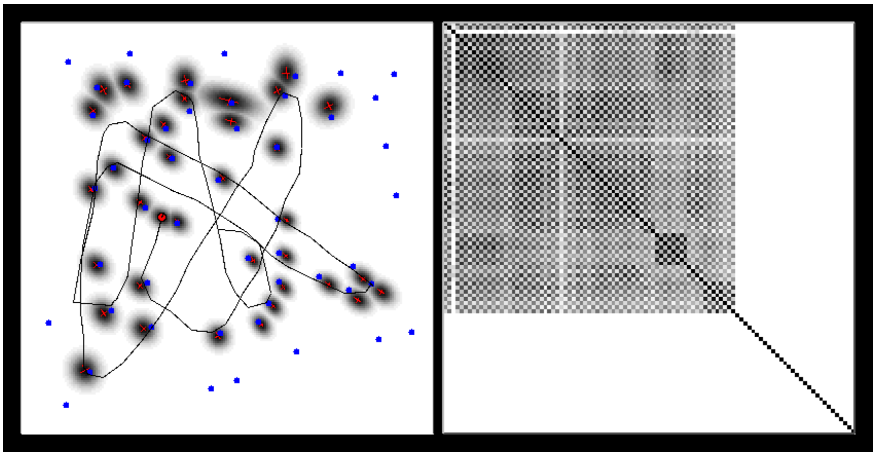
(E)KF-SLAM



Map   Correlation matrix

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

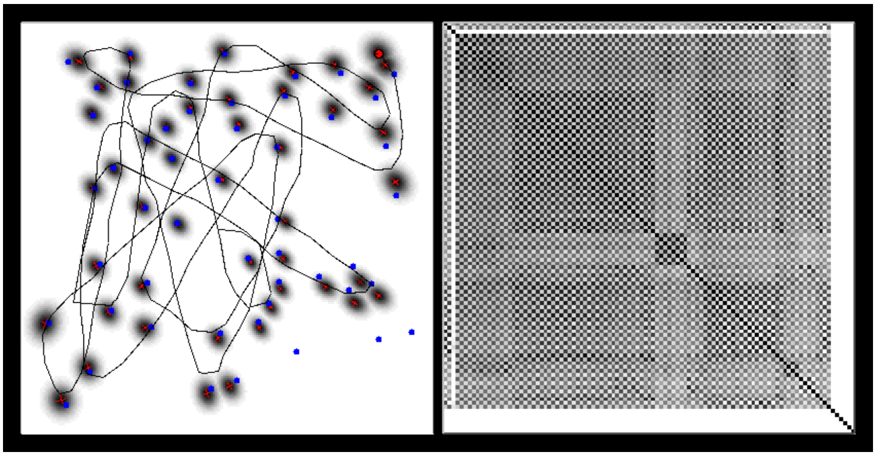
(E)KF-SLAM



Map   Correlation matrix

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

(E)KF-SLAM



Map   Correlation matrix

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

Properties of (E)KF-SLAM (Linear Case)

Theorem [1]:

The determinant of any sub-matrix of the map covariance matrix decreases monotonically as successive observations are made.

Theorem [2]:

In the limit the landmark estimates become fully correlated

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

EKF-SLAM Summary

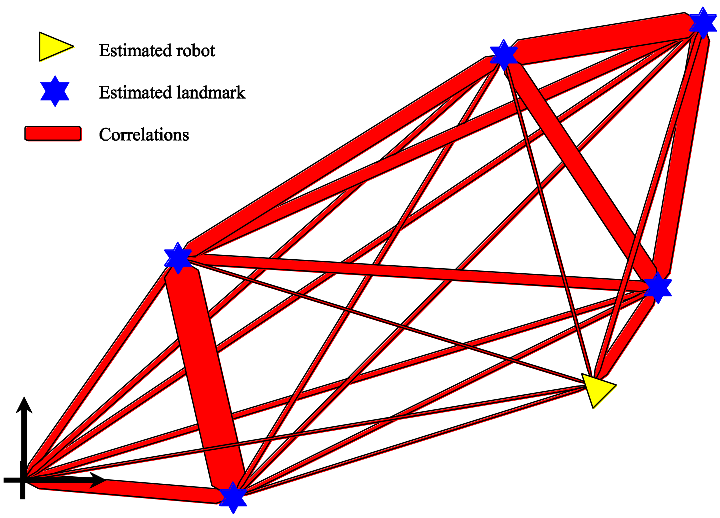
- Quadratic in the number of landmarks: $O(n^2)$
- Convergence results for the linear case.
- Can diverge if nonlinearities are large!
- Have been applied successfully in large-scale environments.
- Approximations reduce the computational complexity.

Source: Burgard, *Probabilistic Robotics*, SLAM Companion Slides

SLAM Convergence

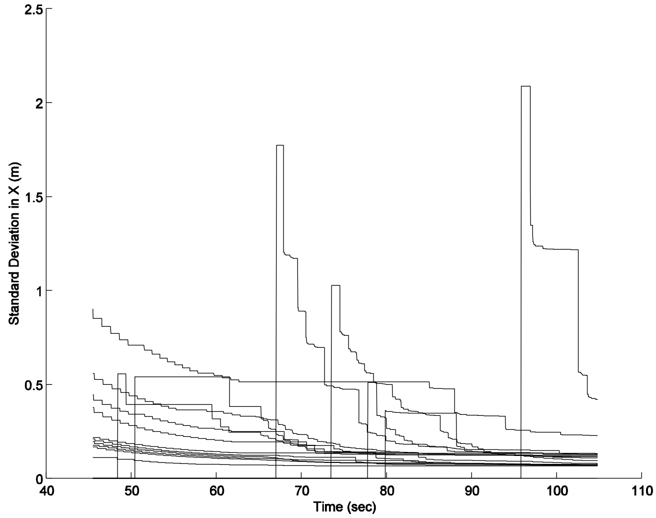
- An observation acts like a displacement to a spring system
  - Effect is greatest in a close neighbourhood
  - Effect on other landmarks diminishes with distance
  - Propagation depends on local stiffness (correlation) properties
- With each new observation the springs become increasingly (and monotonically) stiffer.
- In the limit, a rigid map of landmarks is obtained.
  - A perfect *relative* map of the environment
- The location accuracy of the robot is bounded by
  - The current quality of the map
  - The relative sensor measurement

Spring Analogy



Monotonic Convergence

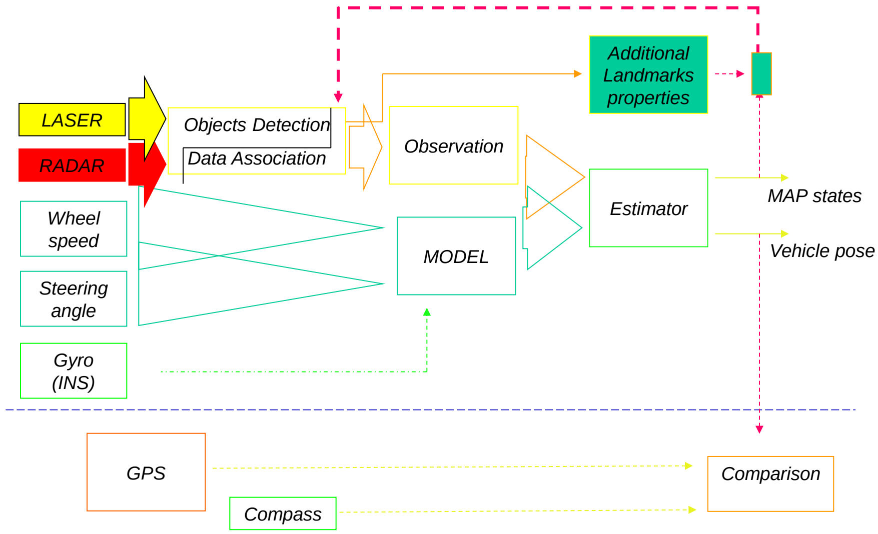
- With each new observation, the determinant decreases over the map and for any submatrix in the map.



Models

- Models are central to creating a representation of the world.
- Must have a mapping between sensed data (eg, laser, cameras, odometry) and the states of interest (eg, vehicle pose, stationary landmarks)
- Two essential model types:
  – Vehicle motion
  – Sensing of external objects

An Example System



States, Controls, Observations

Joint state with momentary pose

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_{v_k} \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_N \end{bmatrix}$$

Joint state with pose history

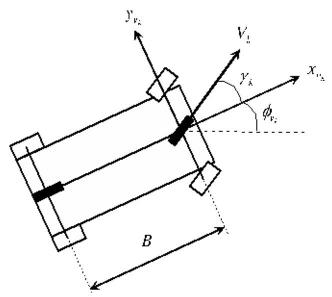
$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_{v_k} \\ \mathbf{x}_{v_{k-1}} \\ \vdots \\ \mathbf{x}_{v_0} \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_N \end{bmatrix}$$

Control inputs: $\mathbf{U}_{0:k} = \{\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_k\} = \{\mathbf{U}_{0:k-1}, \mathbf{u}_k\}$

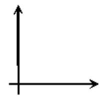
Observations: $\mathbf{Z}_{0:k} = \{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_k\} = \{\mathbf{Z}_{0:k-1}, \mathbf{z}_k\}$

Vehicle Motion Model

- Ackerman steered vehicles: Bicycle model



- Discrete time model:



$$\mathbf{x}_{v_k} = \mathbf{f}_v\left(\mathbf{x}_{v_{k-1}}, \mathbf{u}_k\right) = \begin{bmatrix} x_{v_{k-1}} + V_k \Delta T \cos(\phi_{v_{k-1}} + \gamma_k) \\ y_{v_{k-1}} + V_k \Delta T \sin(\phi_{v_{k-1}} + \gamma_k) \\ \phi_{v_{k-1}} + \frac{V_k \Delta T}{B} \sin(\gamma_k) \end{bmatrix}$$

SLAM Motion Model

$$\mathbf{x}_{v_k} = \mathbf{f}_v\left(\mathbf{x}_{v_{k-1}}, \mathbf{u}_k\right) = \begin{bmatrix} x_{v_{k-1}} + V_k \Delta T \cos(\phi_{v_{k-1}} + \gamma_k) \\ y_{v_{k-1}} + V_k \Delta T \sin(\phi_{v_{k-1}} + \gamma_k) \\ \phi_{v_{k-1}} + \frac{V_k \Delta T}{B} \sin(\gamma_k) \end{bmatrix}$$

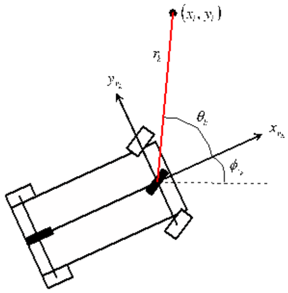
- Joint state: Landmarks are assumed stationary

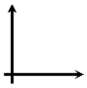
$$\mathbf{x}_k = \begin{bmatrix} \mathbf{f}_v(\mathbf{x}_{v_{k-1}}, \mathbf{u}_k) \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_N \end{bmatrix} \qquad \mathbf{x}_k = \begin{bmatrix} \mathbf{f}_v(\mathbf{x}_{v_{k-1}}, \mathbf{u}_k) \\ \mathbf{x}_{v_{k-1}} \\ \vdots \\ \mathbf{x}_{v_0} \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_N \end{bmatrix}$$

Observation Model

- Range-bearing measurement

$$\mathbf{z}_{i_k} = \mathbf{h}_i(\mathbf{x}_k) = \begin{bmatrix} \sqrt{(x_i - x_{v_k})^2 + (y_i - y_{v_k})^2} \\ \arctan \frac{y_i - y_{v_k}}{x_i - x_{v_k}} - \phi_{v_k} \end{bmatrix}$$





Applying Bayes to SLAM: Available Information

- States $\mathbf{x}_k$ (Hidden or inferred values)
  - Vehicle poses
  - Map; typically composed of discrete parts called landmarks or features
- Controls $\mathbf{U}_{0:k}$
  - Velocity
  - Steering angle
- Observations $\mathbf{Z}_{0:k}$
  - Range-bearing measurements

Augmentation: Adding new poses and landmarks

- Add new pose

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{f}_v(\mathbf{x}_{v_{k-1}}, \mathbf{u}_k) \\ \mathbf{x}_{v_{k-1}} \\ \vdots \\ \mathbf{x}_{v_0} \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_N \end{bmatrix}$$

- Conditional probability is a Markov Model

$$\begin{aligned} p\left(\mathbf{x}_{v_k} | \mathbf{x}_{k-1}\right) &= \int p\left(\mathbf{x}_{v_k} | \mathbf{x}_{k-1}, \mathbf{u}_k\right) p\left(\mathbf{u}_k\right) d\mathbf{u}_k \\ &= \int \delta\left(\mathbf{x}_{v_k} - \mathbf{f}_v\left(\mathbf{x}_{v_{k-1}}, \mathbf{u}_k\right)\right) p\left(\mathbf{u}_k\right) d\mathbf{u}_k \\ &= p\left(\mathbf{x}_{v_k} | \mathbf{x}_{v_{k-1}}\right) \end{aligned}$$

Augmentation

$$\begin{aligned} p\left(\mathbf{x}_{v_k} | \mathbf{x}_{k-1}\right) &= \int p\left(\mathbf{x}_{v_k} | \mathbf{x}_{k-1}, \mathbf{u}_k\right) p\left(\mathbf{u}_k\right) d\mathbf{u}_k \\ &= \int \delta\left(\mathbf{x}_{v_k} - \mathbf{f}_v\left(\mathbf{x}_{v_{k-1}}, \mathbf{u}_k\right)\right) p\left(\mathbf{u}_k\right) d\mathbf{u}_k \\ &= p\left(\mathbf{x}_{v_k} | \mathbf{x}_{v_{k-1}}\right) \end{aligned}$$

- Product rule to create joint PDF $p(x_k)$

$$p\left(\mathbf{x}_{v_k}, \mathbf{x}_{k-1}\right) = p\left(\mathbf{x}_{v_k} | \mathbf{x}_{v_{k-1}}\right) p\left(\mathbf{x}_{v_{k-1}}, \ldots, \mathbf{x}_{v_0}, \mathbf{m}_1, \ldots, \mathbf{m}_N\right)$$

- Same method applies to adding new landmark states

Marginalisation:
Removing past poses and obsolete landmarks

- Augmenting with new pose and marginalising the old pose gives the classical SLAM prediction step

$$p\left(\mathbf{x}_{v_k}, \mathbf{m}_1, \ldots, \mathbf{m}_N\right) = \int p\left(\mathbf{x}_{v_k}, \mathbf{x}_{v_{k-1}}, \mathbf{m}_1, \ldots, \mathbf{m}_N\right) d\mathbf{x}_{v_{k-1}}$$

Fusion: Incorporating observation information

- Conditional PDF according to observation model

$$\begin{aligned} p\left(\mathbf{z}_{i_k} | \mathbf{x}_k\right) &= \int p\left(\mathbf{z}_{i_k} | \mathbf{x}_{v_k}, \mathbf{m}_i, \mathbf{r}_k\right) p\left(\mathbf{r}_k\right) d\mathbf{r}_k \\ &= \int \delta\left(\mathbf{z}_{i_k} - \mathbf{h}\left(\mathbf{x}_{v_k}, \mathbf{m}_i, \mathbf{r}_k\right)\right) p\left(\mathbf{r}_k\right) d\mathbf{r}_k \end{aligned}$$

- Bayes update:
  proportional to product of likelihood and prior

$$p\left(\mathbf{x}_k | \mathbf{Z}_{0:k}\right) = \frac{p\left(\mathbf{z}_{i_k} = \mathbf{z}_0 | \mathbf{x}_k\right) p\left(\mathbf{x}_k | \mathbf{Z}_{0:k-1}\right)}{p\left(\mathbf{z}_{i_k} = \mathbf{z}_0\right)}$$

Implementing Probabilistic SLAM

- The problem is that Bayesian operations are intractable in general.
  - General equations are good for analytical derivations, not good for implementation

- We need approximations
  - Linearised Gaussian systems (EKF, UKF, EIF, SAM)
  - Monte Carlo sampling methods (Rao-Blackwellised particle filters)

Structure of SLAM

- Key property of stochastic SLAM
  - Largely a *parameter* estimation problem

- Since the map is stationary
  - No process model, no process noise

- For Gaussian SLAM
  - Uncertainty in each landmark reduces monotonically after landmark initialisation
  - Map converges

- Examine computational consequences of this structure in next session.

Data Association

- Before the Update Stage we need to determine if the feature we are observing is:
  - An old feature
  - A new feature

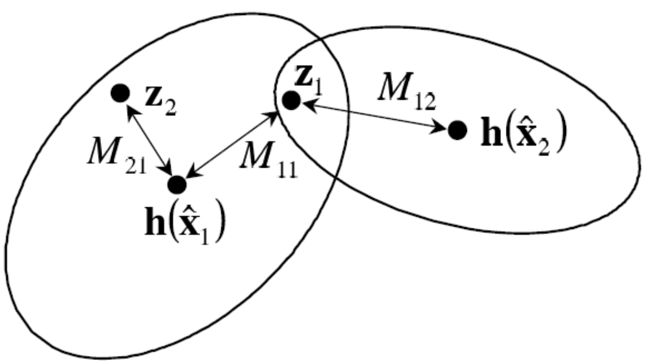
- If there is a match with only one known feature, the Update stage is run with this feature information.

$$\mu(k) = z(k) - h(\hat{x}(k/k-1))$$

$$S(k) = \nabla h_x(k) P(k/k-1) \nabla h_x^T(k) + R$$

$$\alpha = \mu^T(k) S^{-1}(k)\ \mu(k) < \chi^2_{0.95}$$

Validation Gating



New Features

- If there is no match then a potential new feature has been detected
- We do not want to incorporate a spurious observation as a new feature
  - It will not be observed again and will consume computational time and memory
  - It will add clutter, increasing risk of future mis-associations
  - The features are assumed to be static. We don't not want to accept dynamic objects as features: cars, people etc.

Acceptance of New Features: Approach I

- Get the feature in a list of potential features
- Incorporate the feature once it has been observed for a number of times
- Advantages:
  - Simple to implement
  - Appropriate for High Frequency external sensor
- Disadvantages:
  - Loss of information
  - Potentially a problem with sensor with small field of view: a feature may only be seen very few times

Acceptance of New Features: Approach II

- The state vector is extended with past vehicle positions and the estimation of the cross-correlation between current and previous vehicle states is maintained. With this approach improved data association is possible by combining data form various points
  - J. J. Leonard and R. J. Rikoski. *Incorporation of delayed decision making into stochastic mapping*
  - Stephan Williams, PhD Thesis, 2001, University of Sydney
- Advantages:
  - No Loss of Information
  - Well suited to low frequency external sensors (ratio between vehicle velocity and feature rate information)
  - Absolutely necessary for some sensor modalities (eg, range-only, bearing-only)
- Disadvantages:
  - Cost of augmenting state with past poses
  - The implementation is more complicated

Incorporation of New Features

- We have the vehicle states and previous map

$$P_0 = \begin{bmatrix} P_{v,v}^0 & P_{v,m}^0 \\ P_{m,v}^0 & P_{m,m}^0 \end{bmatrix}$$

- We observed a new feature and the covariance and cross-covariance terms need to be evaluated

$$P_1 = \begin{bmatrix} P_{v,v}^0 & P_{v,m}^0 & ? \\ P_{m,v}^0 & P_{m,m}^0 & ? \\ ? & ? & ? \end{bmatrix}$$

Incorporation of New Features: Approach I

$$P_0 = \begin{bmatrix} P_{vv}^0 & P_{vm}^0 & 0 \\ P_{mv}^0 & P_{mm}^0 & 0 \\ 0 & 0 & A \end{bmatrix}$$

With A very large

$$W(k) = P(k/k-1)\nabla h_x^T(k) S^{-1}(k)$$
$$S(k) = \nabla h_x(k) P(k/k-1)\nabla h_x^T(k) + R$$
$$P(k/k) = P(k/k-1) - W(k)S(k)W^T(k)$$

- Easy to understand and implement
- Very large values of A
  may introduce numerical problems

$$P_1 = \begin{bmatrix} P_{vv}^1 & P_{vm}^1 & P_{vn}^1 \\ P_{mv}^1 & P_{mm}^1 & P_{mn}^1 \\ P_{nv}^1 & P_{nm}^1 & P_{nn}^1 \end{bmatrix}$$

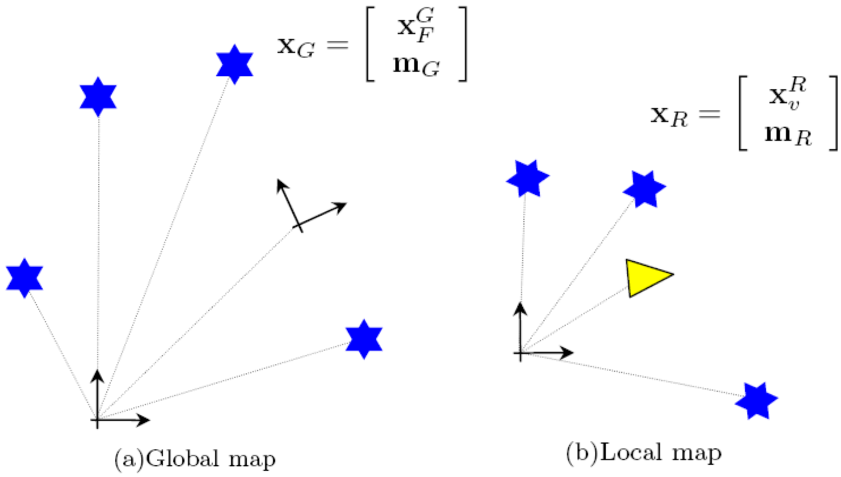
Incorporation of New Features: Analytical Approach

- We can also evaluate the analytical expressions of the new terms

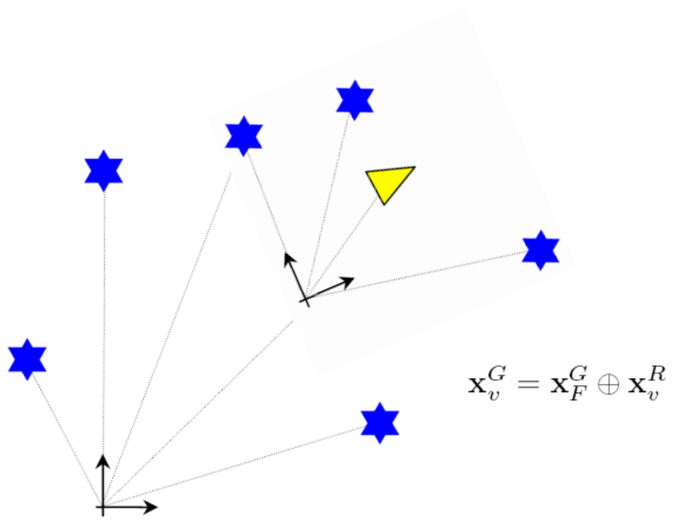
$$P_0 = \begin{bmatrix} P_{v,v}^0 & P_{v,m}^0 \\ P_{m,v}^0 & P_{m,m}^0 \end{bmatrix}$$

$$P_1 = \begin{bmatrix} P_{v,v}^0 & P_{v,m}^0 & ? \\ P_{m,v}^0 & P_{m,m}^0 & ? \\ ? & ? & ? \end{bmatrix}$$

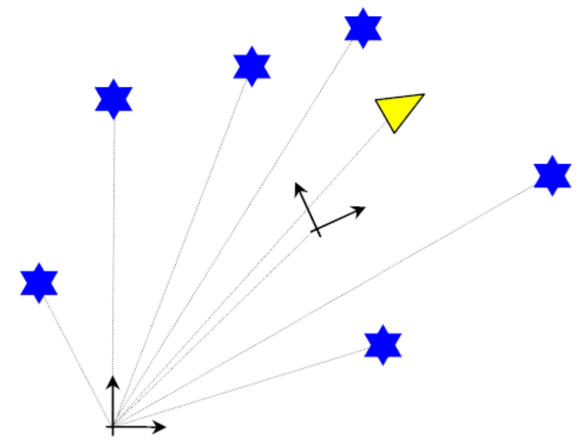
Constrained Local Submap Filter



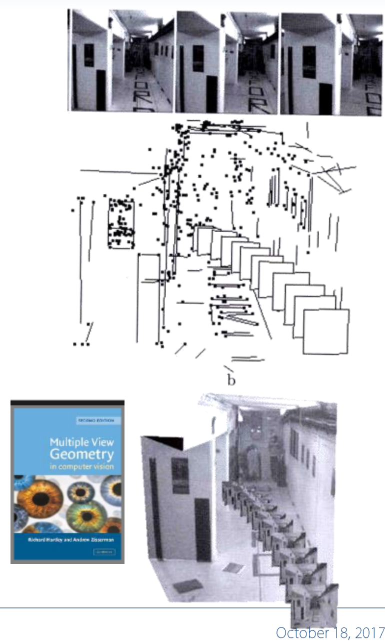
CLSF Registration



CLSF Global Estimate



SLAM: 30+ Year History!



Source: Leonard (MIT) Hartley and Zisserman, Cambridge University Press, p. 437

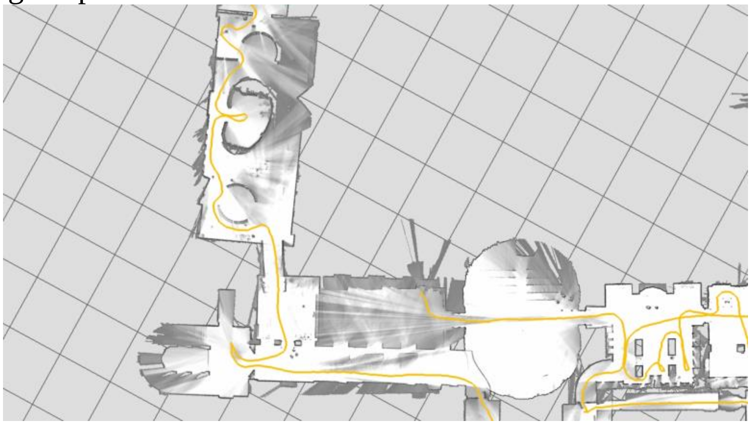
Jenkin Building Basement, Circa 1989

Source: Leonard (MIT)

Cool Robotics Share

Cartographer

- Google Open Source SLAM



https://opensource.googleblog.com/2016/10/introducing-cartographer.html