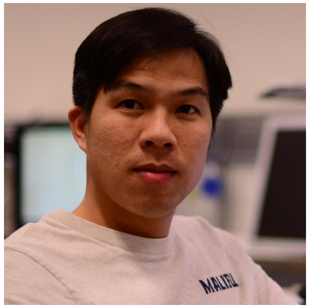
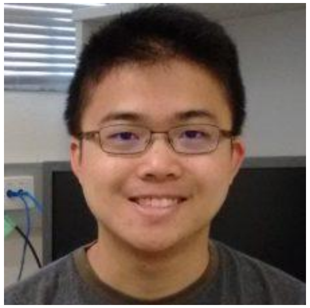
RVSS 2018 : Semantic SLAM
Making robots map and understand the world

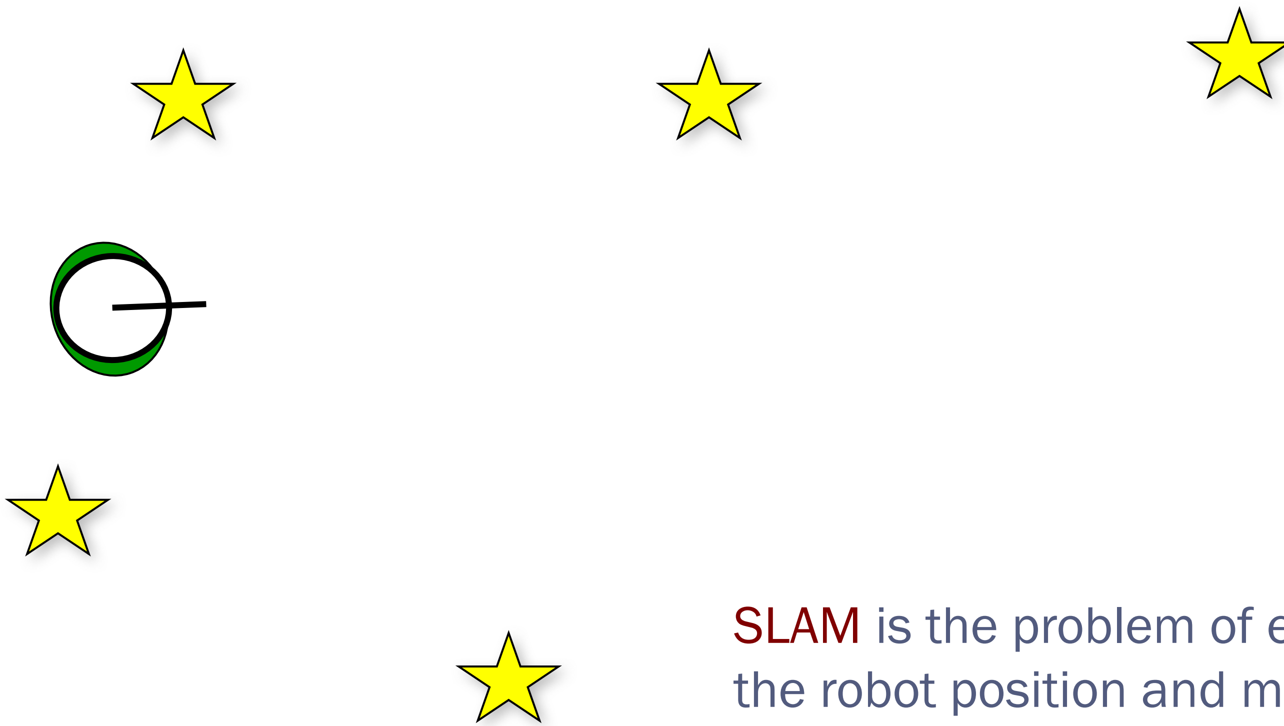
Viorela Ila, Yasir Latif ,Trung Pham, Vincent Lui



SLAM

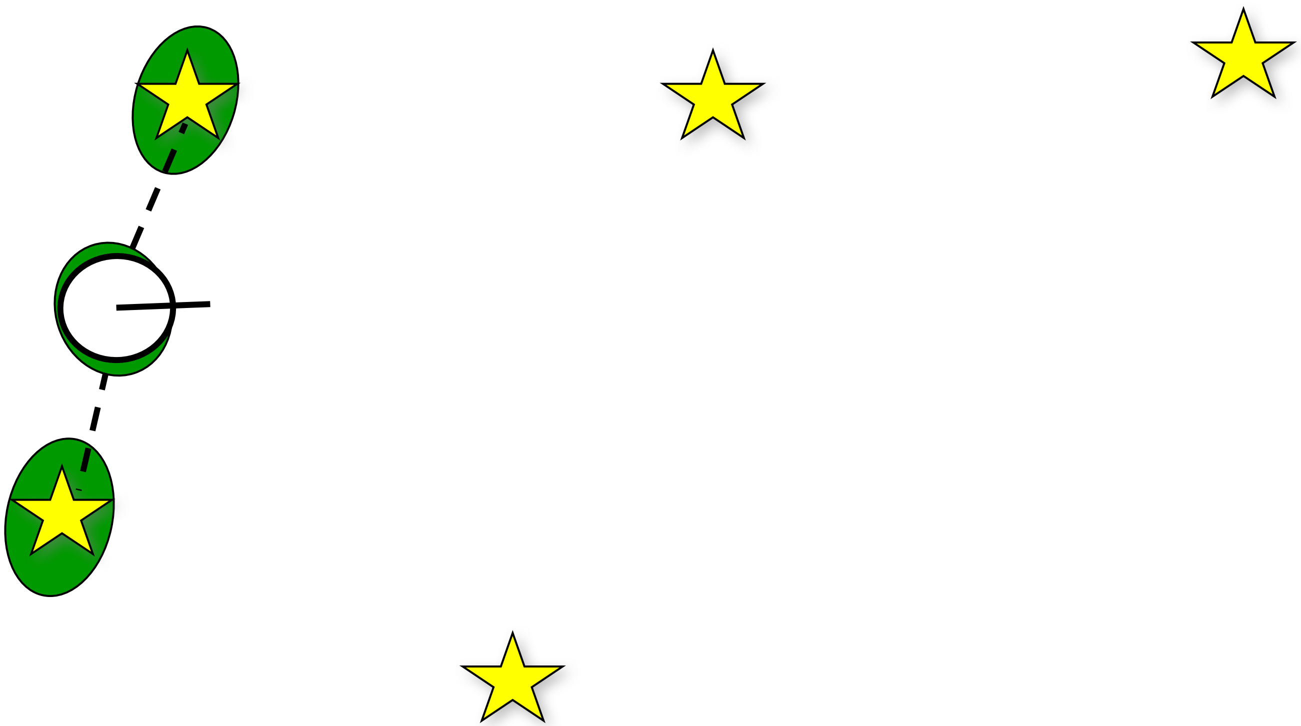
Starts from known position but unknown environment



SLAM is the problem of estimating the robot position and map the environment given the sensor data and control inputs.

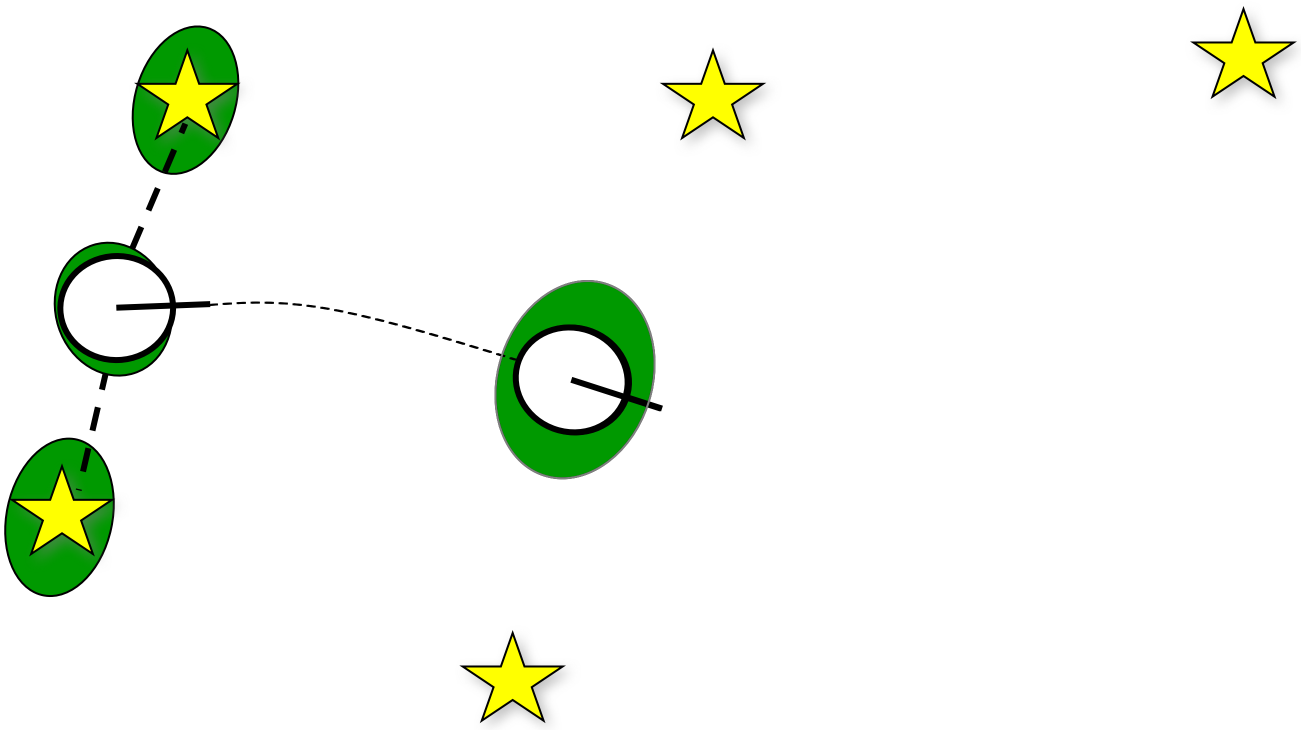
SLAM

Observes landmarks in the environment



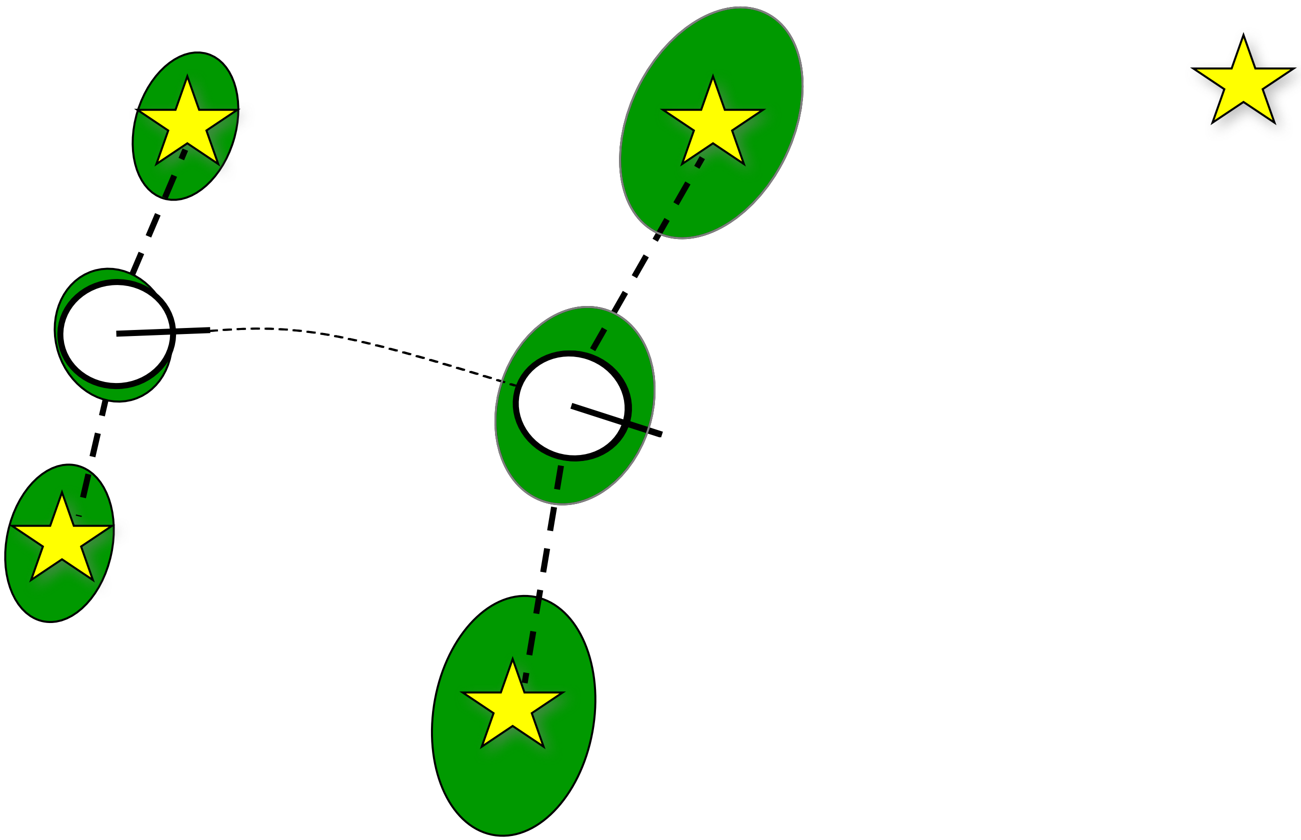
SLAM

Move in the environment.



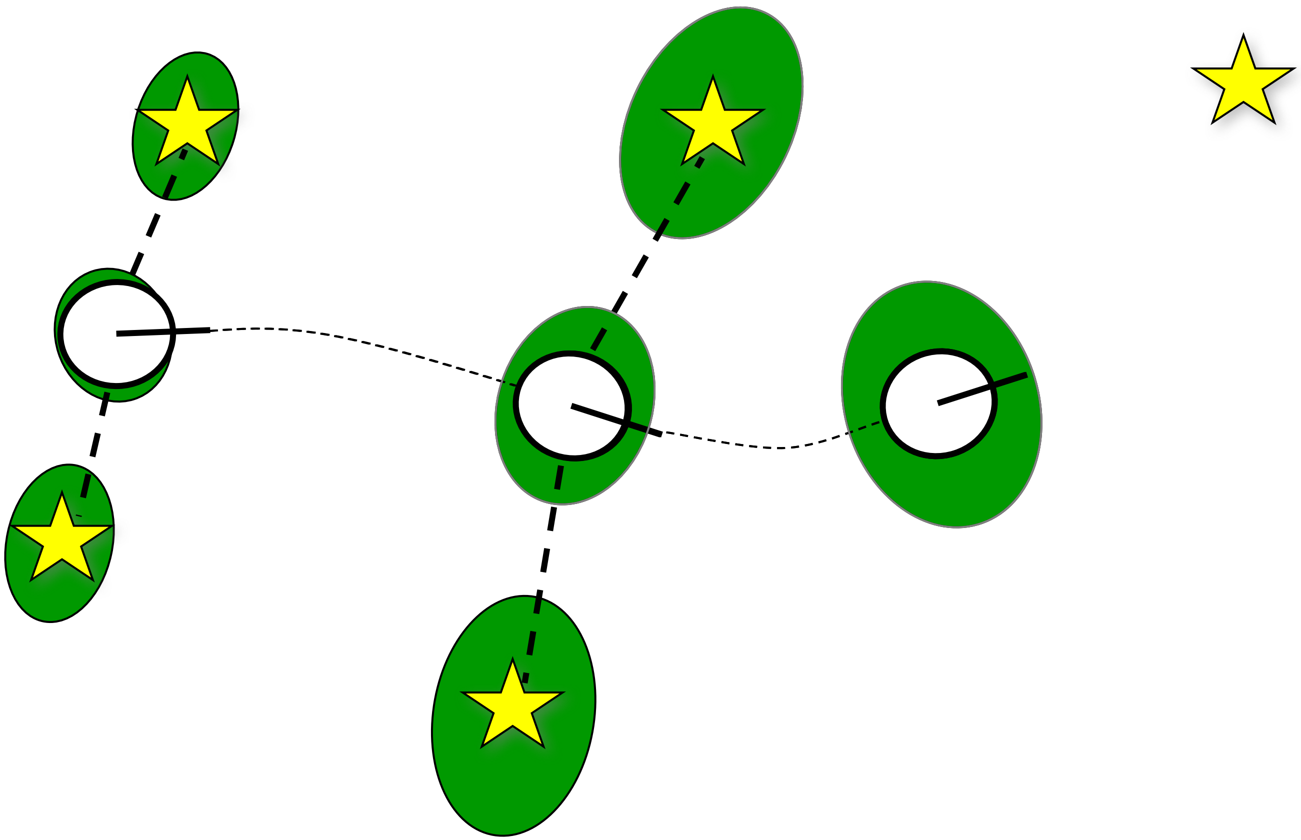
SLAM

Observes landmarks in the environment



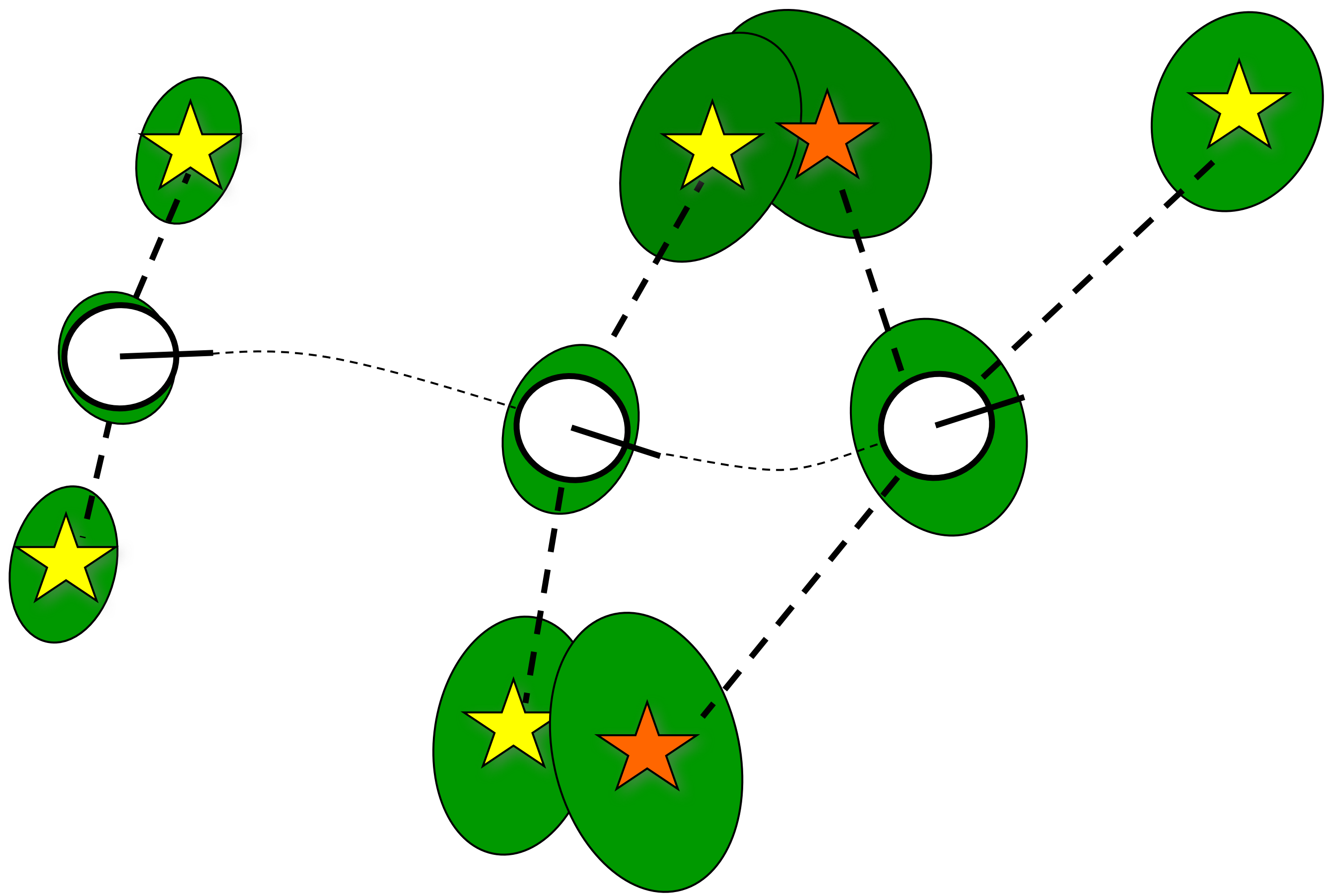
SLAM

Move again.



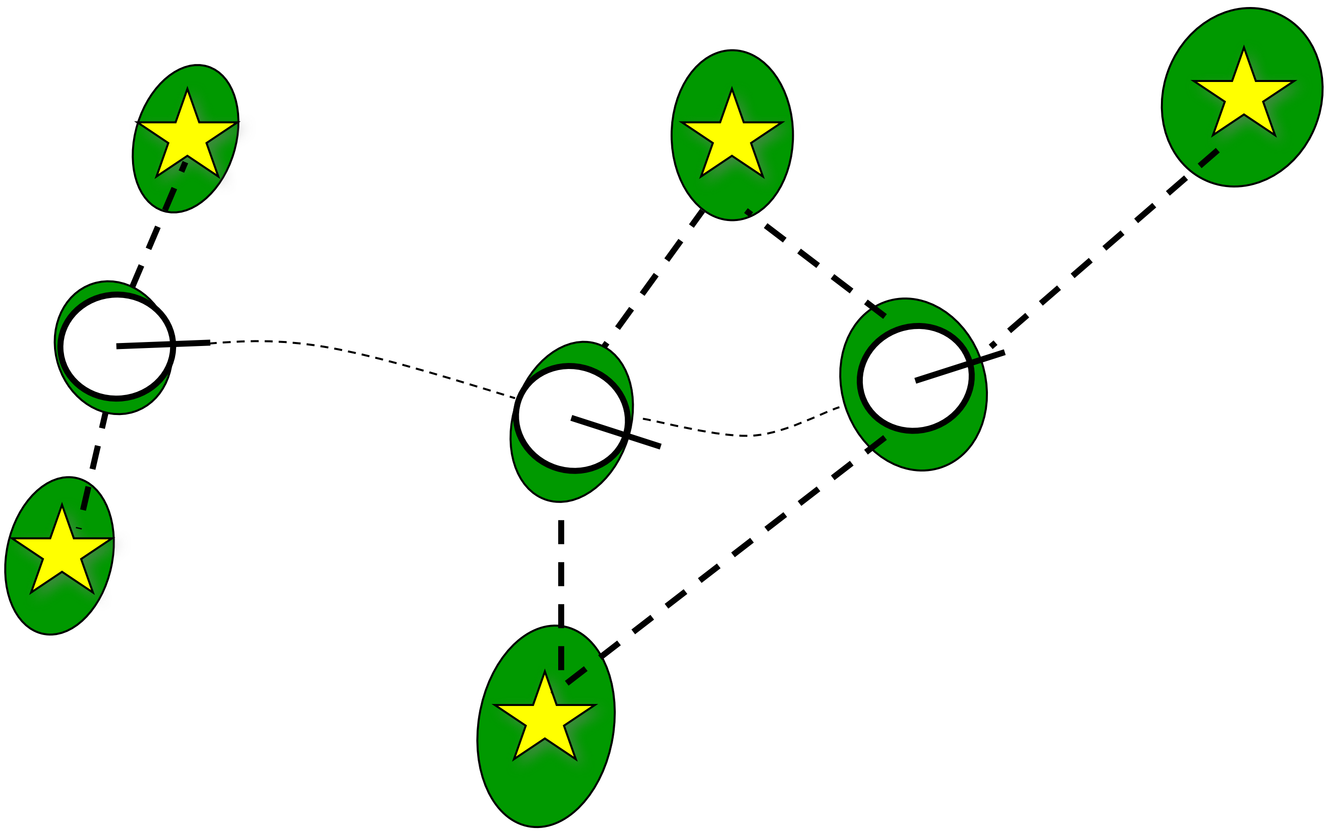
SLAM

Re-observe landmarks in the environment – Data association



SLAM

Re-observe landmarks in the environment – Reduces the error



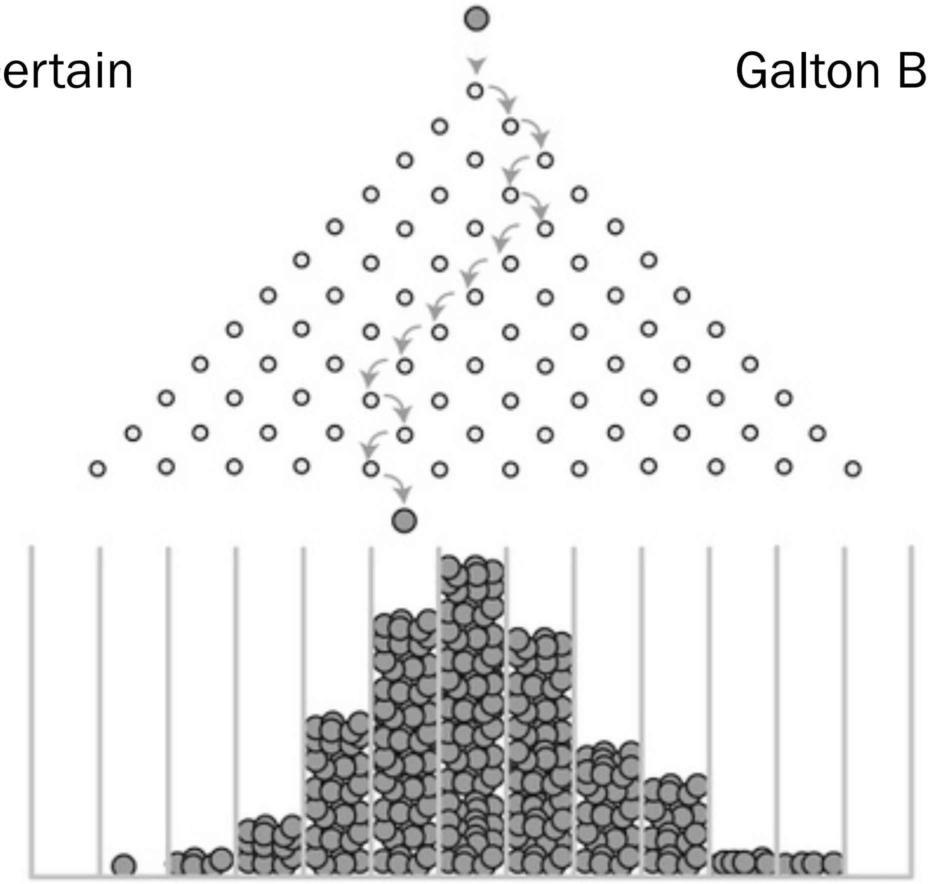
Challenges of Robot SLAM

- Most mobile robots work in an **unstructured, uncertain environment.**
- Absolute position information (e.g. via GPS or other global localization systems such as VICON) is often unavailable, inaccurate, or insufficient
- **Uncertainties** are present in sensors readings, motion as well as in the model.
  - Sensor noise
  - Sensor aliasing
  - Effecter/Actuator noise
  - Position integration
  - Simple models

Probability and Gaussian

Example of un uncertain physical process.

Galton Board

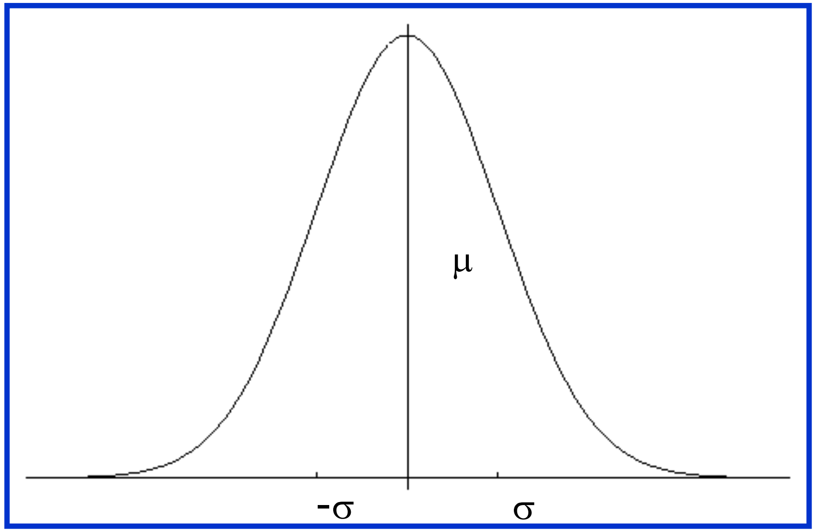


Gaussian

Univariate

$$p(x) \sim N(\mu, \sigma^2):$$

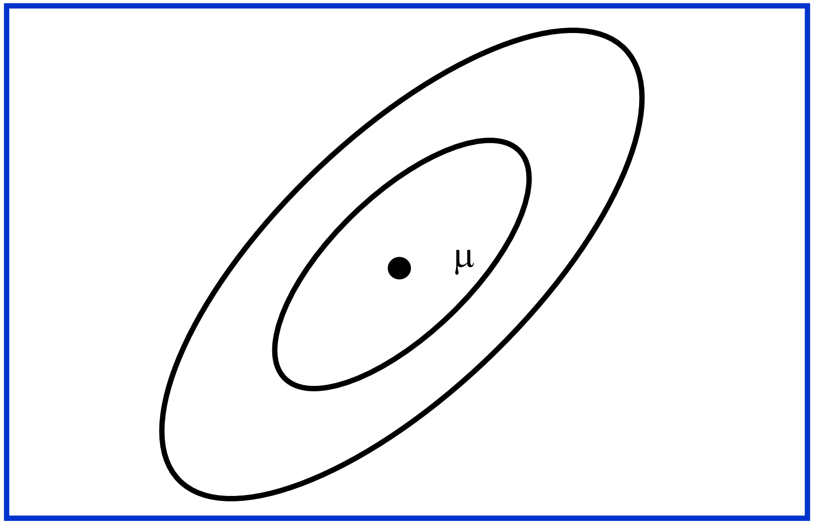
$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$



Multivariate

$$p(\mathbf{x}) \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma}):$$

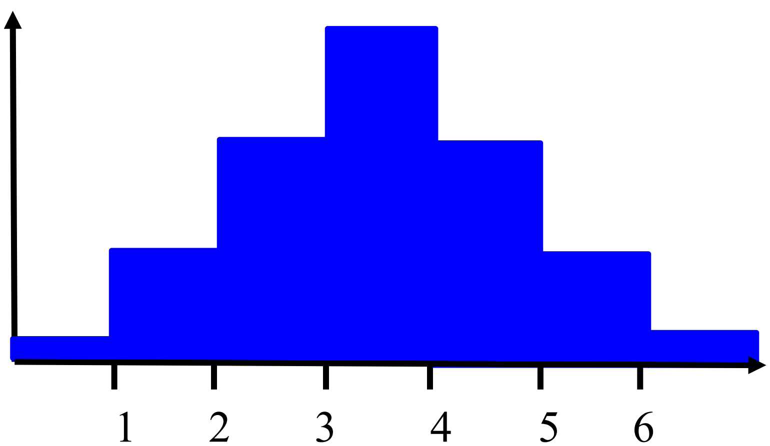
$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$



Discrete vs. Continuous

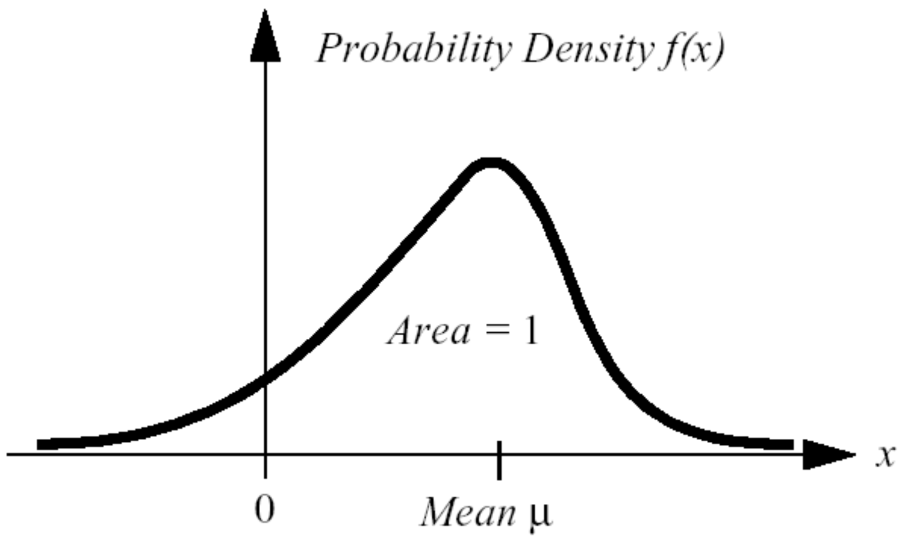
Discrete case

$$\sum_x P(x) = 1$$



Continuous case

$$\int p(x)\, dx = 1$$



Probabilities - Terminology

- Uncertainty → random variable → probability p(x)
- Probability density function pdf p(x)

- Joint probability p(x,y)
- Conditional probability or posterior p(x|y)
- Marginal probability or prior p(x)

Localization

1D world represented with cells

belief	x_1	x_2	x_3	x_4	x_5

What is the probability of the robot being in a cell?

1/5	**1/5**	**1/5**	**1/5**	**1/5**

$$p(x') = 0.2$$

Measurement

The robot sees the red color

x_1	x_2	x_3	x_4	x_5

$$p(z = red) = 0.6$$

$$p(z = white) = 0.2$$

How this affect the probability distribution (belief) ?

0.04	0.12	0.12	0.04	0.04

$\neq 1$

Multiply each cell with the probability of being red or white

Measurement

0.04	**0.12**	**0.12**	**0.04**	**0.04**

$$\sum_i p(z \mid x_i') p(x_i') = 0.36$$

To have a valid probability we need to normalize:

1/9	**1/3**	**1/3**	**1/9**	**1/9**

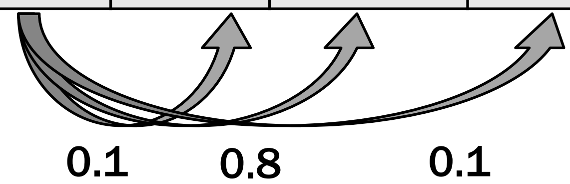
Posterior distribution $p(x \mid z)$

Noisy Motion

The robot is moving 2 cells to the right. The world is cyclic.

0	0.5	0.5	0	0

$$u = 2$$



$$p(x_{i+2} \mid x_i) = 0.8$$

$$p(x_{i+1} \mid x_i) = 0.1$$

$$p(x_{i+3} \mid x_i) = 0.1$$

0.1*0+ 0.8*0+ 0.1*0.5	0.1*0+ 0.8*0+ 0.1*0	0.1*0.5+ 0.8*0+ 0.1*0	0.1*0.5+ 0.8*0.5+ 0.1*0	0.1*0+ 0.8*0.5+ 0.1*0.5

0.05	0	0.05	0.45	0.45

Sensing and Motion

Sensing ➔ Bayes rule ➔ Posterior

$$p(x_i \mid z) = \eta p(z \mid x_i) p(x_i)$$

Motion ➔ Total probability ➔ Prior

$$p(x_i^t) = \sum_j p(x_j^{t-1}) p(x_i \mid x_j)$$

Markov Localization

- Named after Russian mathematician Andrey Markov
- Applies to any type of distribution

Prediction

$$\overline{bel}(x_t) = \int_x p(x_t \mid u_t; x_{t-1})\, bel(x_{t-1}) dx_{t-1}$$

- continuous

$$\overline{bel}(x_t) = \sum_x p(x_t \mid u_t; x_{t-1})\, bel(x_{t-1})$$

- discrete

Update – calculates the posterior

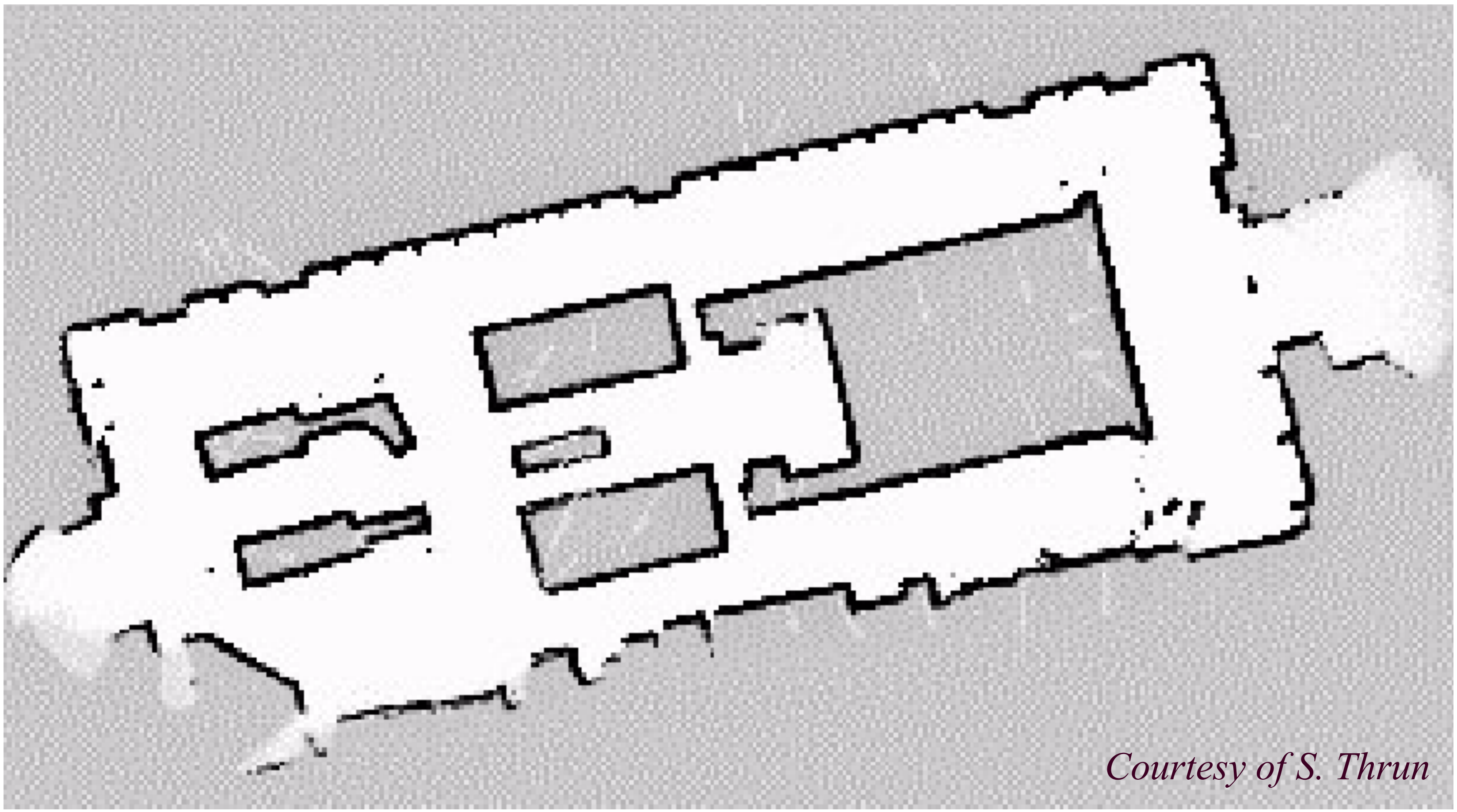
$$bel(x_t) = \eta p(z \mid x_t)\, \overline{bel}(x_{t-1})$$

Mapping – Grid Mapping

Workshop Material

Occupancy grid representation

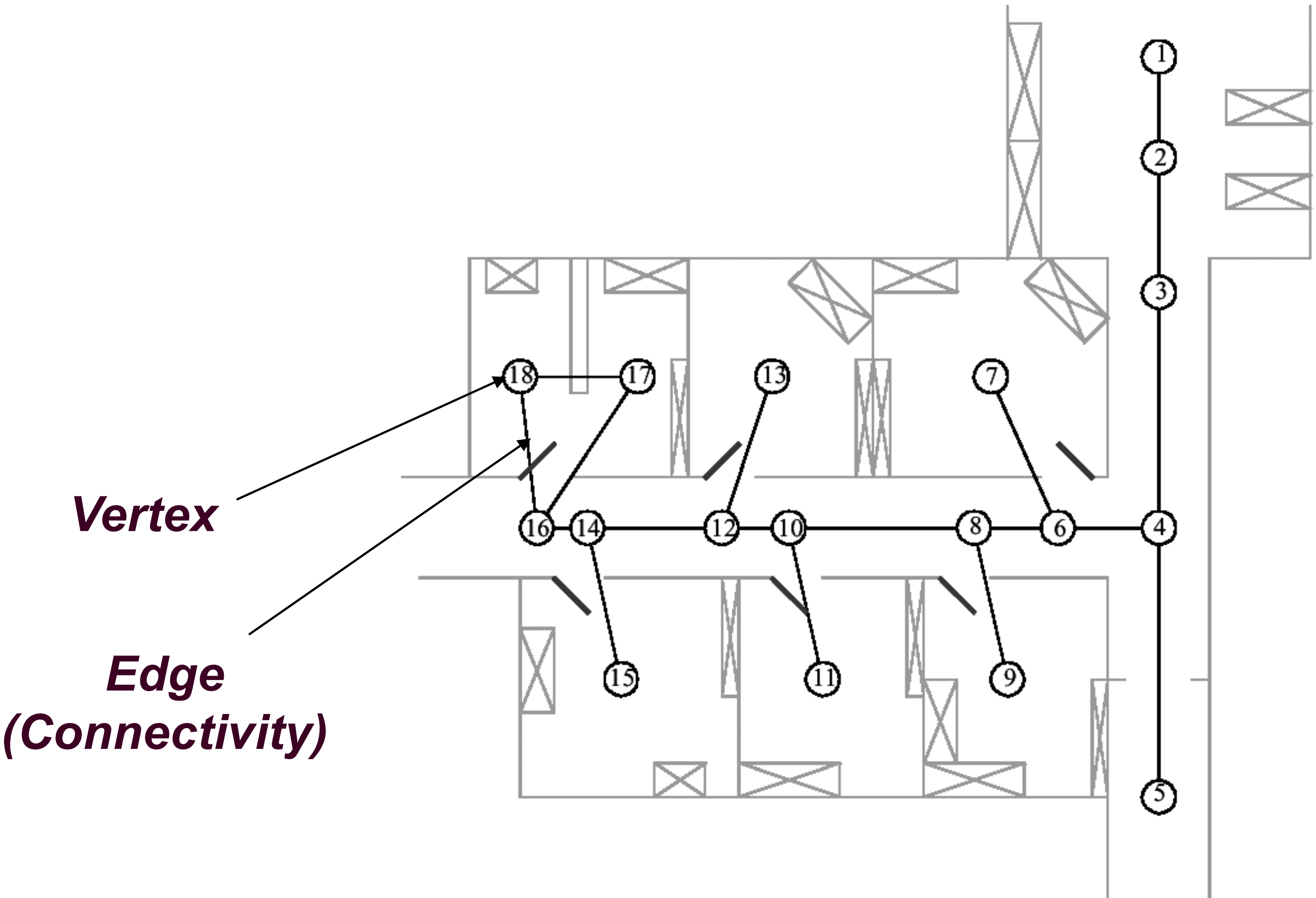
- Fixed cell decomposition – Example with very small cells



Occupancy grid representation

- The map is explicitly given.
- Objects are visible. Object avoidance can be applied.
- Path planning can be easily applied.
  - Small cells: extract straight lines, define polygons, find the configuration space, plan a path.
  - Large cells: Apply search algorithms directly in the cell space
- Restricted to small, structured environments

Topological Map Representation

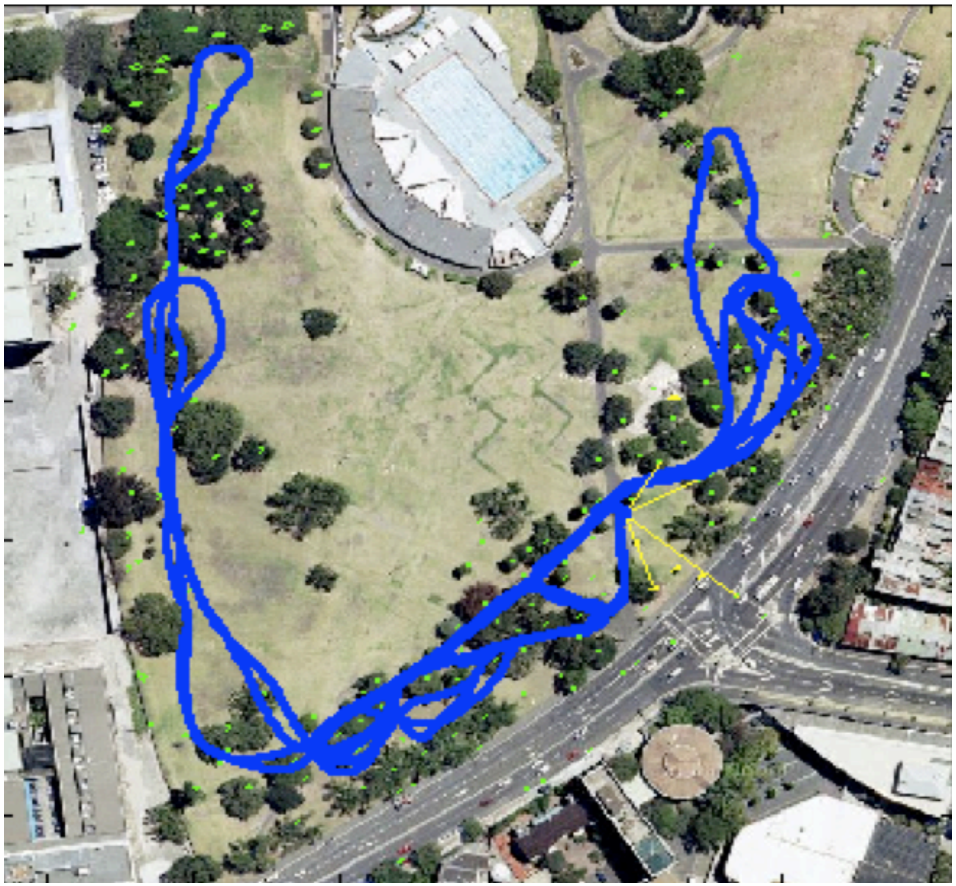


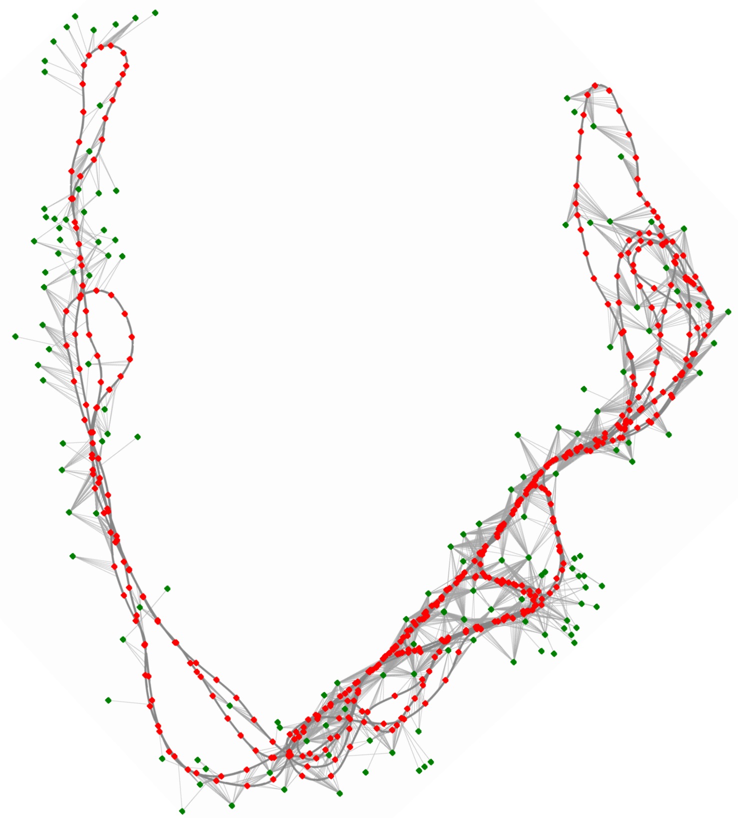
Topological Map Representation

- The map is not explicitly given.
- Separate object detection and avoidance need to be performed in order to safely navigate.
- Path planning is done by search algorithms in a graph.
- Can be used in large, unstructured environments.
- The map can be obtained by drawing the exterioceptive sensor readings (laser scans, 2D/3D points, objects) for each vertex.

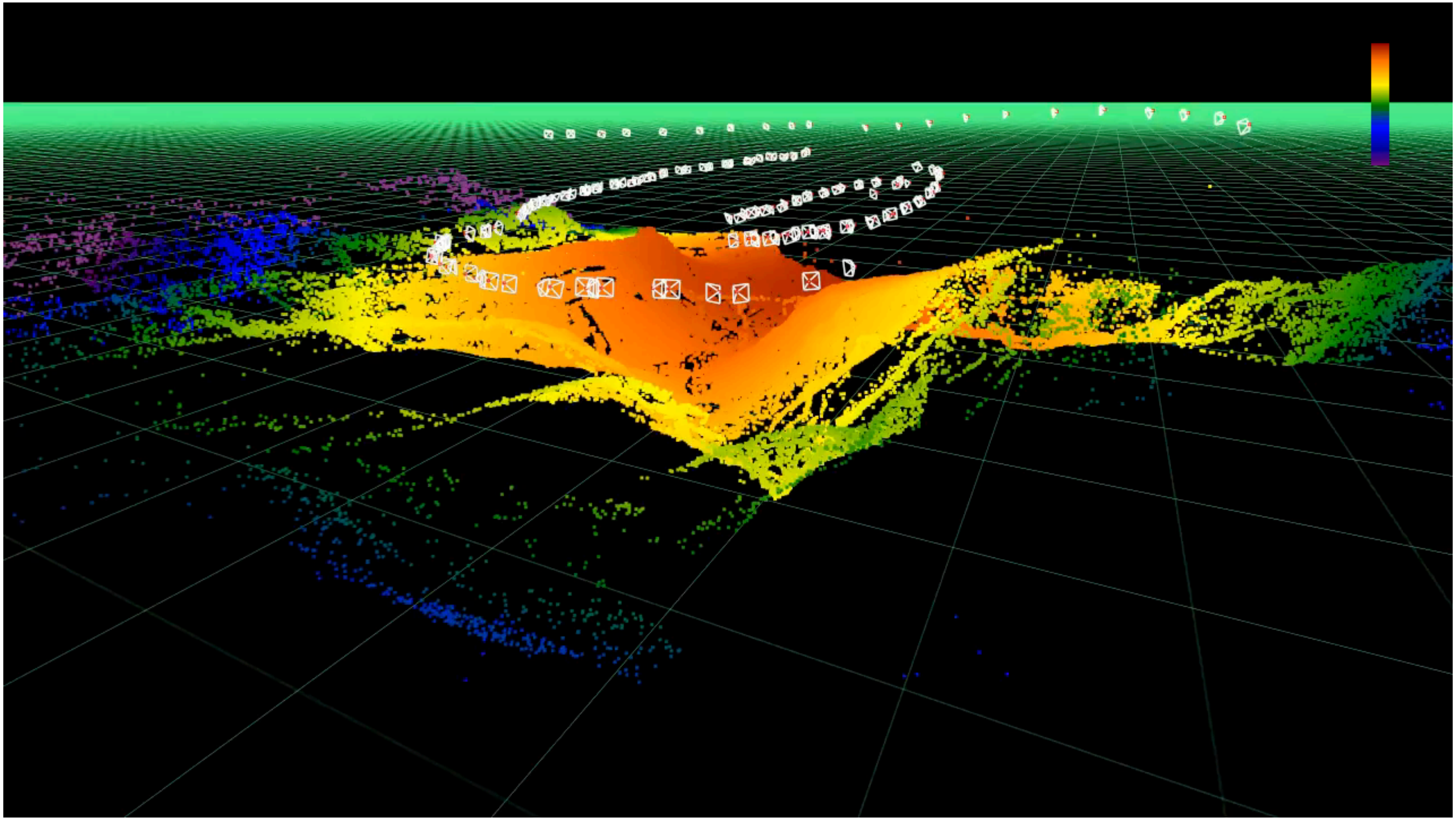
Laser scans-based maps

2D Landmark-based Maps

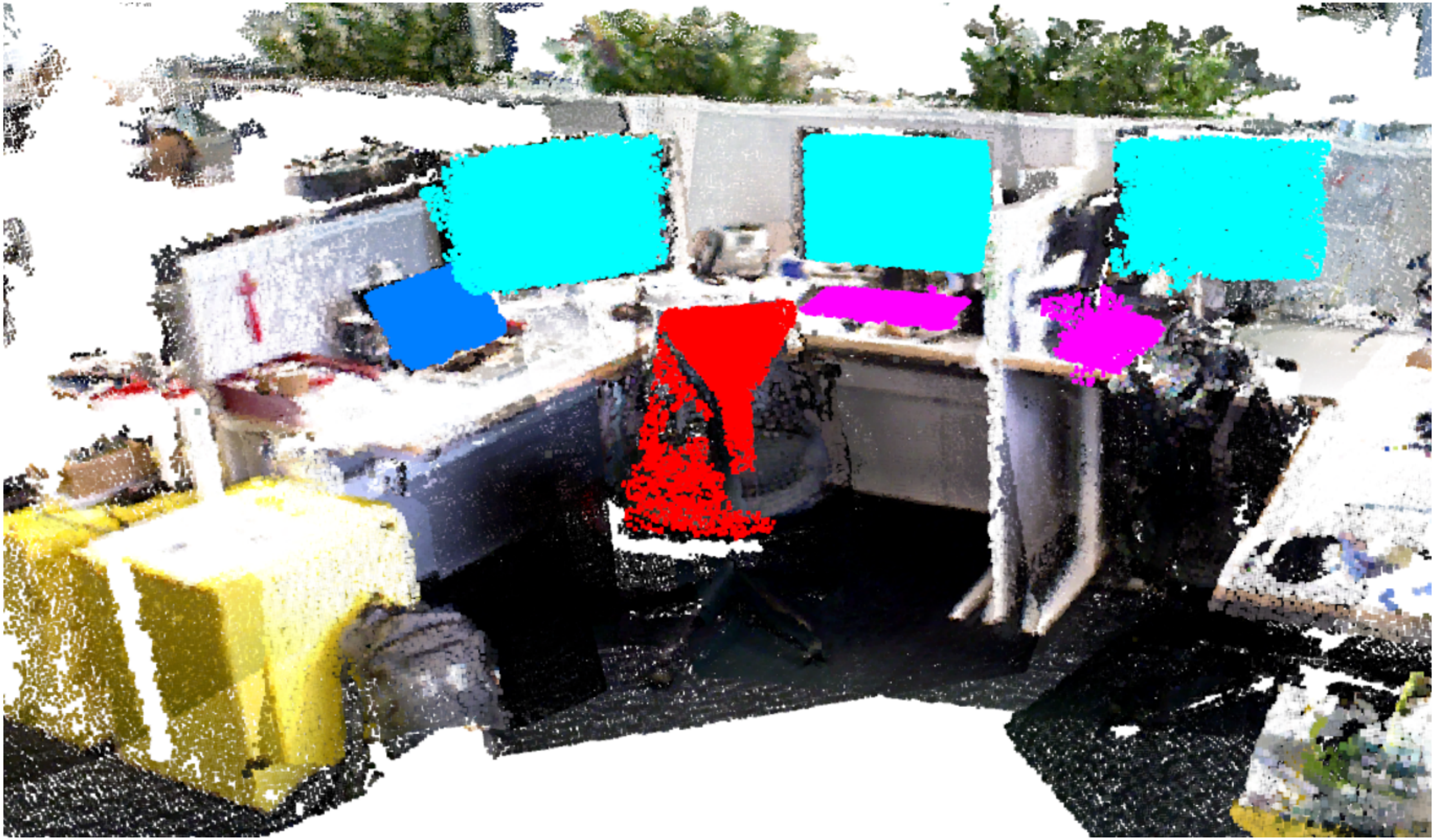




3D Points from 2D Image Processing

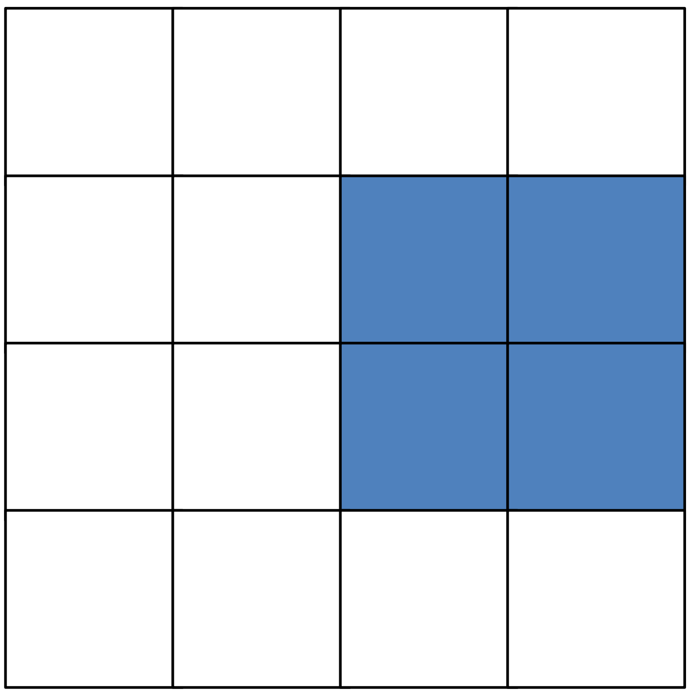


Dense-Semantic SLAM



Grid Maps

Occupancy grid maps address the problem of generating consistent maps from noisy and uncertain measurement data, under the assumption that the robot pose is known.



Unoccupied

Occupied

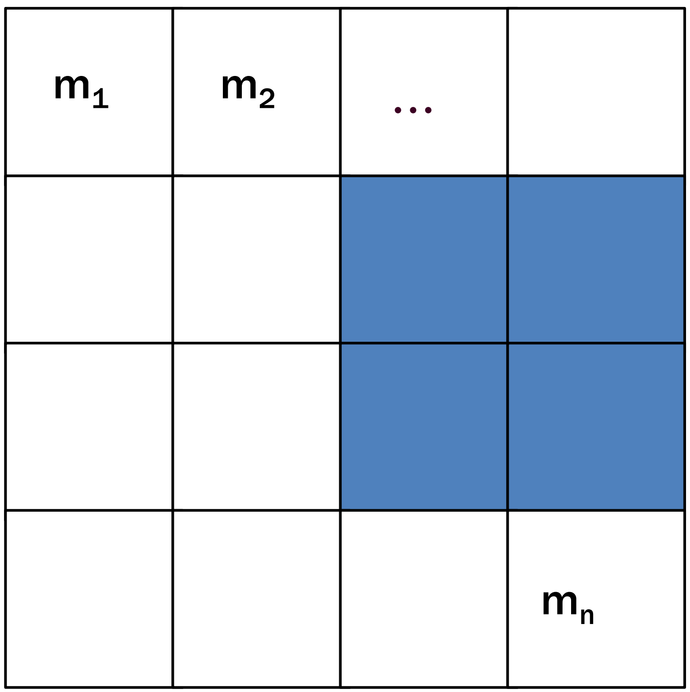
Each random variable is binary and corresponds to the occupancy of the location is covers

Represent the map as a field of random variables, arranged in an evenly spaced grid.

Grid Maps – Representation

$$\{m_1, m_2, \ldots, m_n)$$

- Random variables



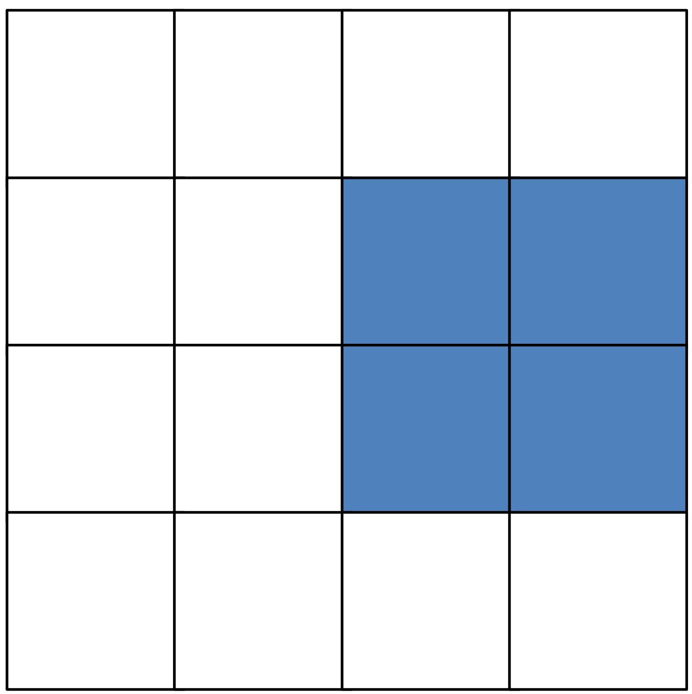
Unoccupied $p(m_j) = 0$

Occupied $p(m_i) = 1$

Unknown $p(m_j) = 0.5$

Grid Maps – Assumptions

- Assumes the environment is static



Unoccupied

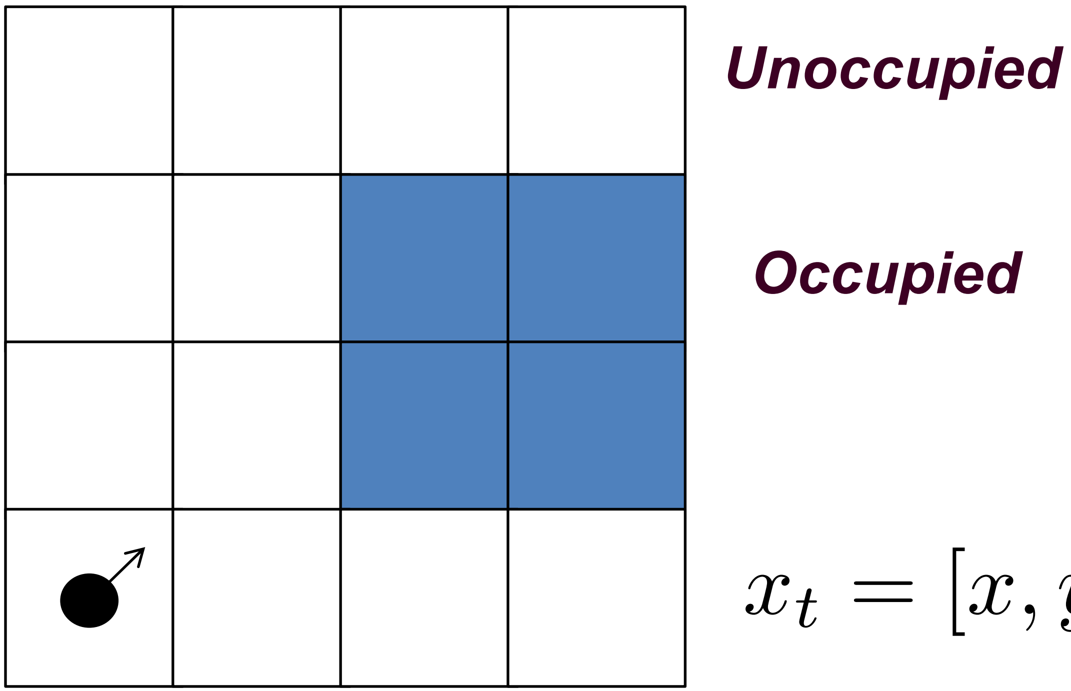
- Always

Occupied

- Always

Grid Maps – Assumptions

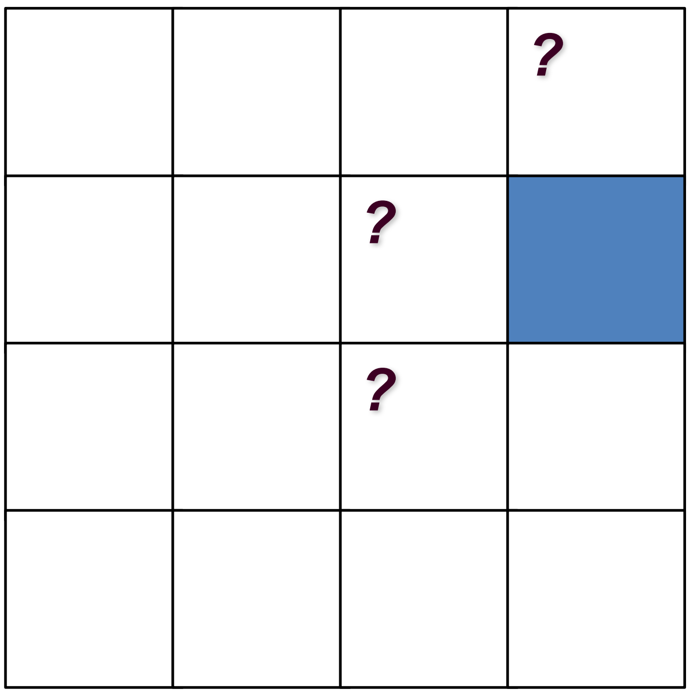
- Assumes known robot position and orientation



$$x_t = [x, y, \theta]^\top$$

Grid Maps – Assumptions

- Independent cells : If I know part of the environment does not help in estimating the rest



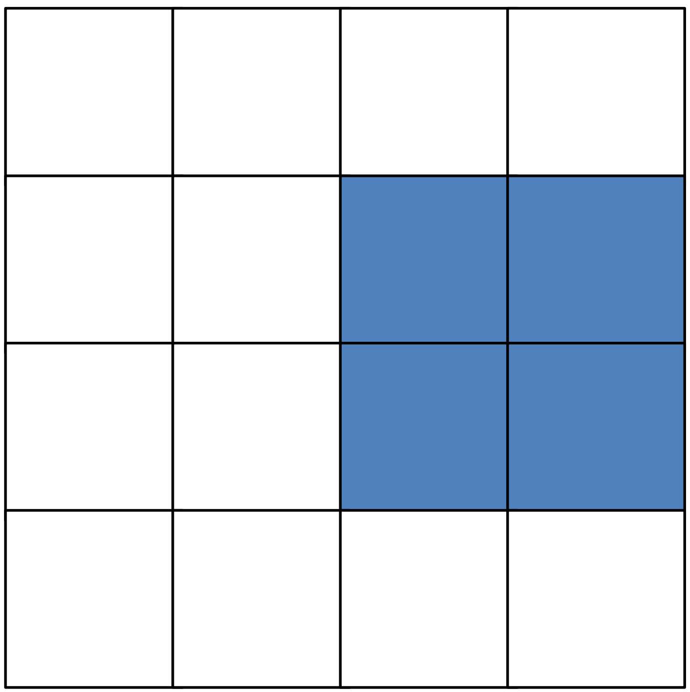
Occupied

$$p(m_i) = 1$$

Grid Maps – Representation

$$\mathbf{m} = \{m_1, m_2, \ldots, m_n)$$

- Independent random variables



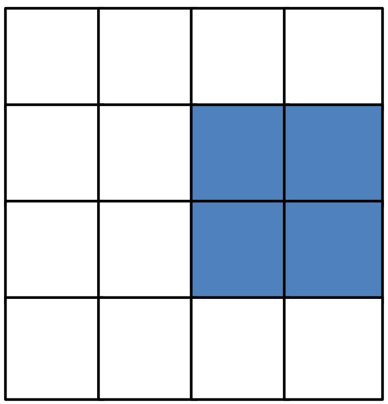
$$p(\mathbf{m}) = \prod_{i=1}^{n} p(m_i)$$

The problem can be broken-down into a collection of separate problem.

Grid Maps – Representation

$$\mathbf{m} = \{m_1, m_2, \dots, m_n\}$$

- Independent random variables



$$\mathbf{z}_{1:t}$$

- All measurements

$$\mathbf{x}_{1:t}$$

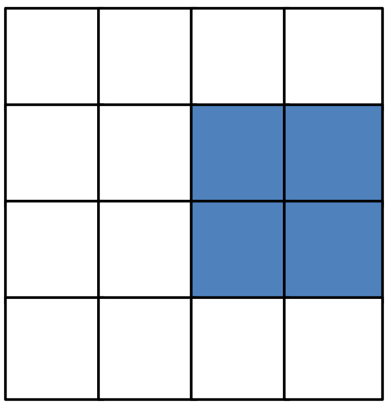
- Robot poses

Mapping assumes known robot position

$$p(\mathbf{m}|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \prod_{i=1}^{n} p(m_i|\mathbf{z}_{1:t}, \mathbf{x}_{1:t})$$

Grid Maps – Bayes Filter

- Apply Bayes filter for mapping



$\mathbf{z}_{1:t}$

- All measurements

$\mathbf{x}_{1:t}$

- Robot poses

We don't have actions ➔ no Prediction step

$$p(\mathbf{m}|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \prod_{i=1}^{n} p(m_i|\mathbf{z}_{1:t}, \mathbf{x}_{1:t})$$

Grid Maps – Bayes Filter – Update

- Apply Bayes rule to calculate the probability of each cell given the current measurements and the poses of the robot.

$$p(m_i|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(z_t|m_i, \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})\, p(m_i|\mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}{p(z_t|\mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

Grid Maps – Bayes Filter – Update

- Apply Bayes rule to calculate the probability of each cell given the current measurements and the poses of the robot.

Measurement probability ***Current belief***

$$p(m_i|\mathbf{z}_{1:t},\mathbf{x}_{1:t}) = \frac{p(z_t|m_i,\mathbf{z}_{1:t-1},\mathbf{x}_{1:t})\, p(m_i|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t})}{p(z_t|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t})}$$

Evidence

Grid Maps – Bayes Filter – Update

Let's integrate all the assumptions in our Posterior calculation:

$$p(m_i | \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(z_t | m_i, \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})\, p(m_i | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}{p(z_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

Markov assumption

$$p(m_i | \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(z_t | m_i, \mathbf{x}_t)\, p(m_i | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(z_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

Given the map, the current measurement does not depend on previous poses and measurements.

Grid Maps – Bayes Filter – Update

$$p(m_i|\mathbf{z}_{1:t},\mathbf{x}_{1:t}) = \frac{p(z_t|m_i,\mathbf{x}_t)\, p(m_i|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t-1})}{p(z_t|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t})}$$

Bayes rule again

$$p(z_t|m_i,\mathbf{x}_t) = \frac{p(m_i|z_t,\mathbf{x}_t)\, p(z_t|\mathbf{x}_t)}{p(m_i|\mathbf{x}_t)}$$

$$p(m_i|\mathbf{z}_{1:t},\mathbf{x}_{1:t}) = \frac{p(m_i|z_t,\mathbf{x}_t)\, p(z_t|\mathbf{x}_t)\, p(m_i|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t-1})}{p(m_i|\mathbf{x}_t)\, p(z_t|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t})}$$

Grid Maps – Bayes Filter – Update

Let's integrate all the assumptions in our Posterior calculation:

$$p(m_i|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(m_i|z_t, \mathbf{x}_t)\, p(z_t|\mathbf{x}_t)\, p(m_i|\mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(m_i)\, p(z_t|\mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

We have binary states:

$$p(\neg m_i|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(\neg m_i|z_t, \mathbf{x}_t)\, p(z_t|\mathbf{x}_t)\, p(\neg m_i|\mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\neg m_i)\, p(z_t|\mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

$$\frac{p(m_i|\mathbf{z}_{1:t}, \mathbf{x}_{1:t})}{p(\neg m_i|\mathbf{z}_{1:t}, \mathbf{x}_{1:t})} = \frac{p(m_i|\mathbf{z}_{1:t}, \mathbf{x}_{1:t})}{1 - p(m_i|\mathbf{z}_{1:t}, \mathbf{x}_{1:t})}$$

Grid Maps – Bayes Filter – Update

Let's integrate all the assumptions in our Posterior calculation:

$$p(m_i | \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(m_i | z_t, \mathbf{x}_t)\; p(z_t | \mathbf{x}_t)\; p(m_i | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(m_i)\; p(z_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

We have binary states:

$$p(\neg m_i | \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(\neg m_i | z_t, \mathbf{x}_t)\; p(z_t | \mathbf{x}_t)\; p(\neg m_i | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\neg m_i)\; p(z_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

$$\frac{p(m_i | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})}{p(\neg m_i | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})} = \frac{p(m_i | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})}{1 - p(m_i | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})}$$

Grid Maps – Update

$$\frac{p(m_i|\mathbf{z}_{1:t},\mathbf{x}_{1:t})}{p(\neg m_i|\mathbf{z}_{1:t},\mathbf{x}_{1:t})} = \frac{p(m_i|z_t,\mathbf{x}_t)}{p(\neg m_i|z_t,\mathbf{x}_t)} \frac{p(m_i|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t-1})}{p(\neg m_i|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t-1})} \frac{p(\neg m_i)}{p(m_i)}$$

$$\frac{p(m_i|\mathbf{z}_{1:t},\mathbf{x}_{1:t})}{p(\neg m_i|\mathbf{z}_{1:t},\mathbf{x}_{1:t})} = \frac{p(m_i|z_t,\mathbf{x}_t)}{1-p(m_i|z_t,\mathbf{x}_t)} \frac{p(m_i|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t-1})}{1-p(m_i|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t-1})} \frac{1-p(m_i)}{p(m_i)}$$

Current observation — *Recursive term* — *Prior*

Odds and Log Odds

Odds

$$\frac{p(x)}{p(\neg x)} = \frac{p(x)}{1 - p(x)}$$

$$\frac{p(x)}{1 - p(x)} = y(x)$$

$$p(x) = y(x) - y(x)p(x)$$

$$p(x) = \frac{y(x)}{1 + y(x)} = \frac{1}{1 + \frac{1}{y(x)}}$$

Take the log()

$$p(x) = (1 + y(x)^{-1})^{-1}$$

Odds and Log Odds

Odds

$$\frac{p(x)}{p(\neg x)} = \frac{p(x)}{1 - p(x)}$$

Log Odds

$$l(x) = \log \frac{p(x)}{1 - p(x)}$$

$$p(x) = 1 - \frac{1}{1 + \exp(l(x))}$$

Grid Maps – Odds and Log Odds

Odds

$$\frac{p(m_i|\mathbf{z}_{1:t},\mathbf{x}_{1:t})}{p(\neg m_i|\mathbf{z}_{1:t},\mathbf{x}_{1:t})} = \frac{p(m_i|z_t,\mathbf{x}_t)}{1-p(m_i|z_t,\mathbf{x}_t)} \frac{p(m_i|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t-1})}{1-p(m_i|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t-1})} \frac{1-p(m_i)}{p(m_i)}$$

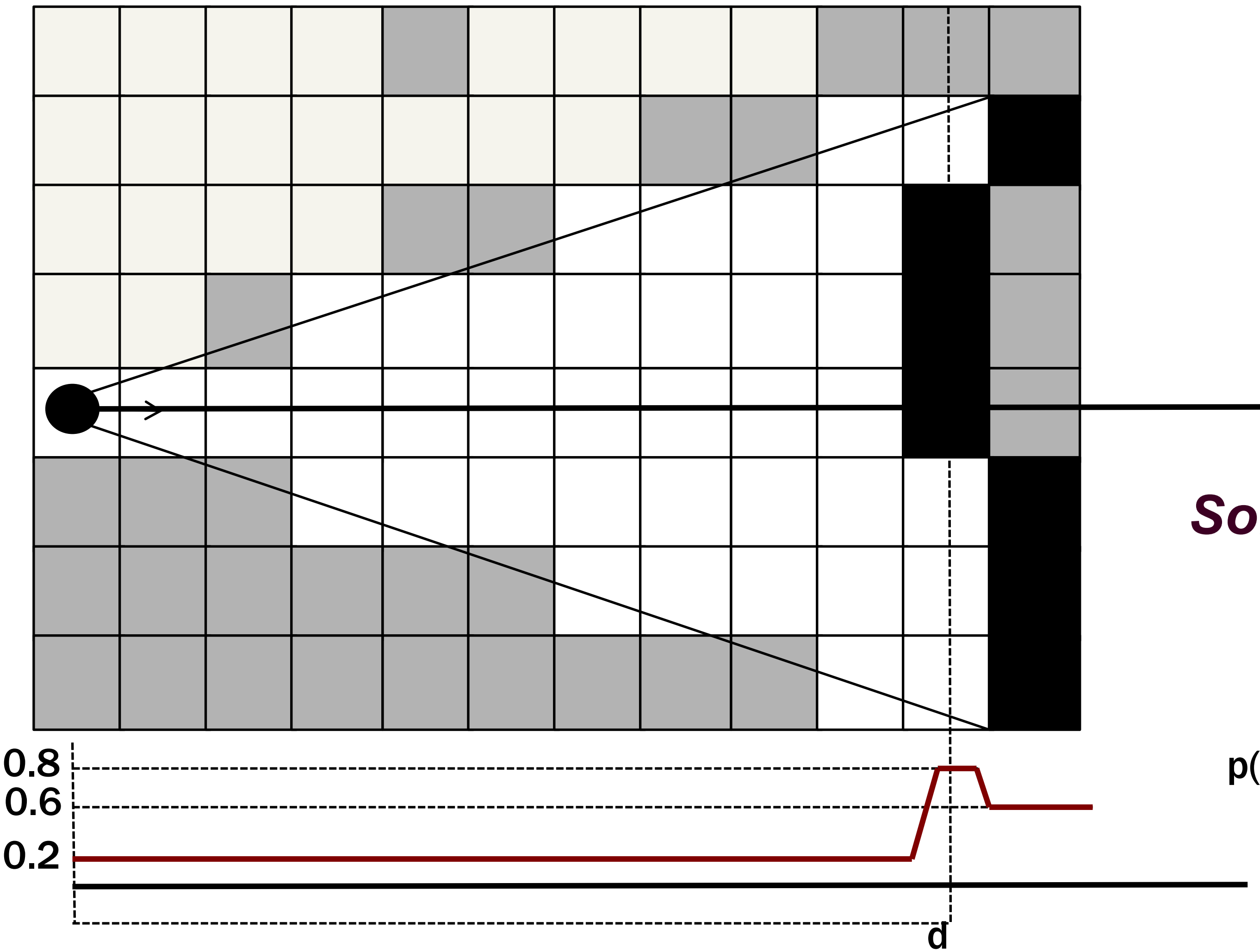
Current observation *Recursive term* *Prior*

Log Odds

$$l(m_i|\mathbf{z}_{1:t},\mathbf{x}_{1:t}) = l(m_i|z_t,\mathbf{x}_t) + l(m_i|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t-1}) - l(m_i)$$

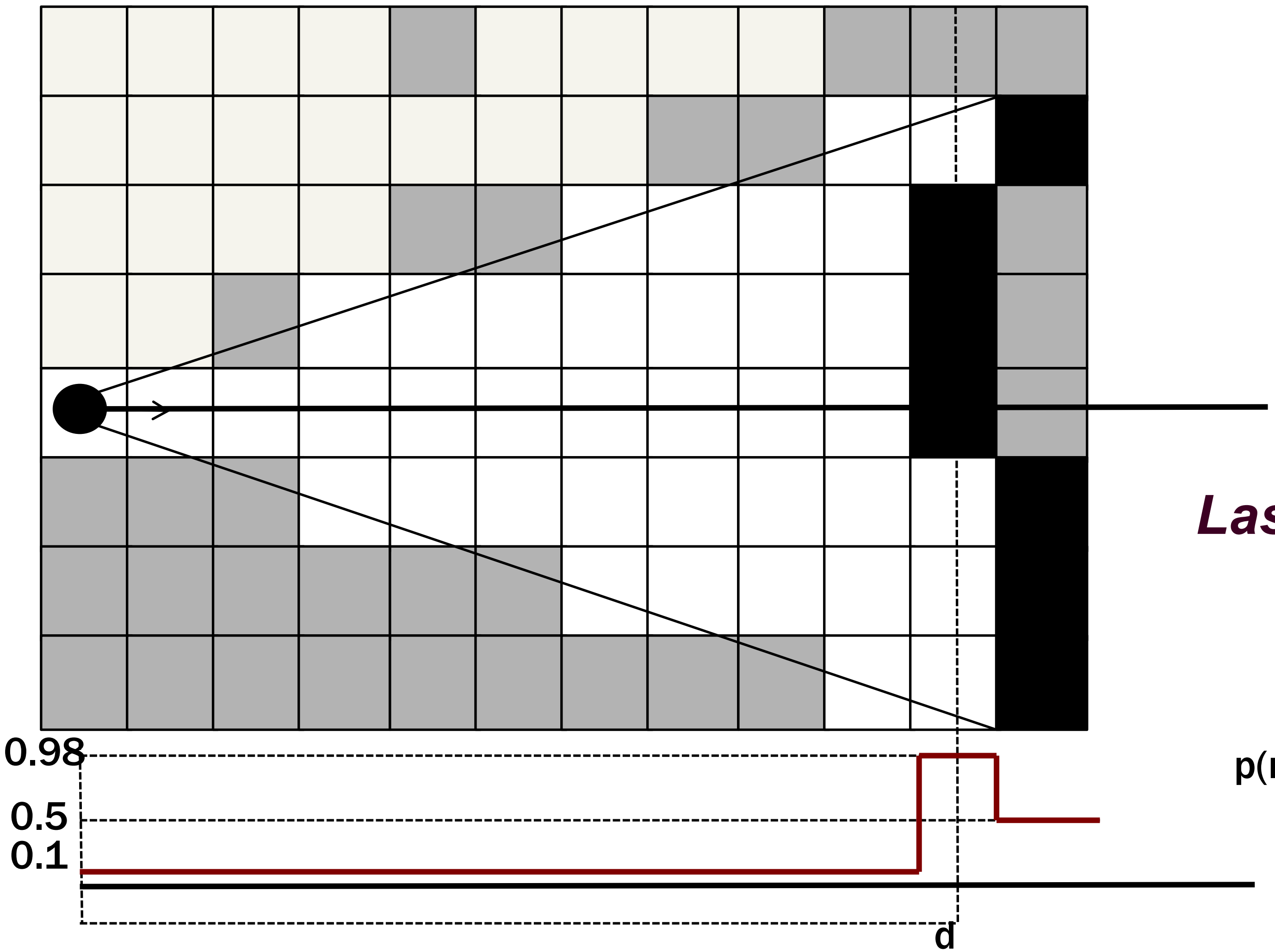
Inverse Sensor Model

Inverse Sensor Model



Sonar example

Inverse Sensor Model



Laser example

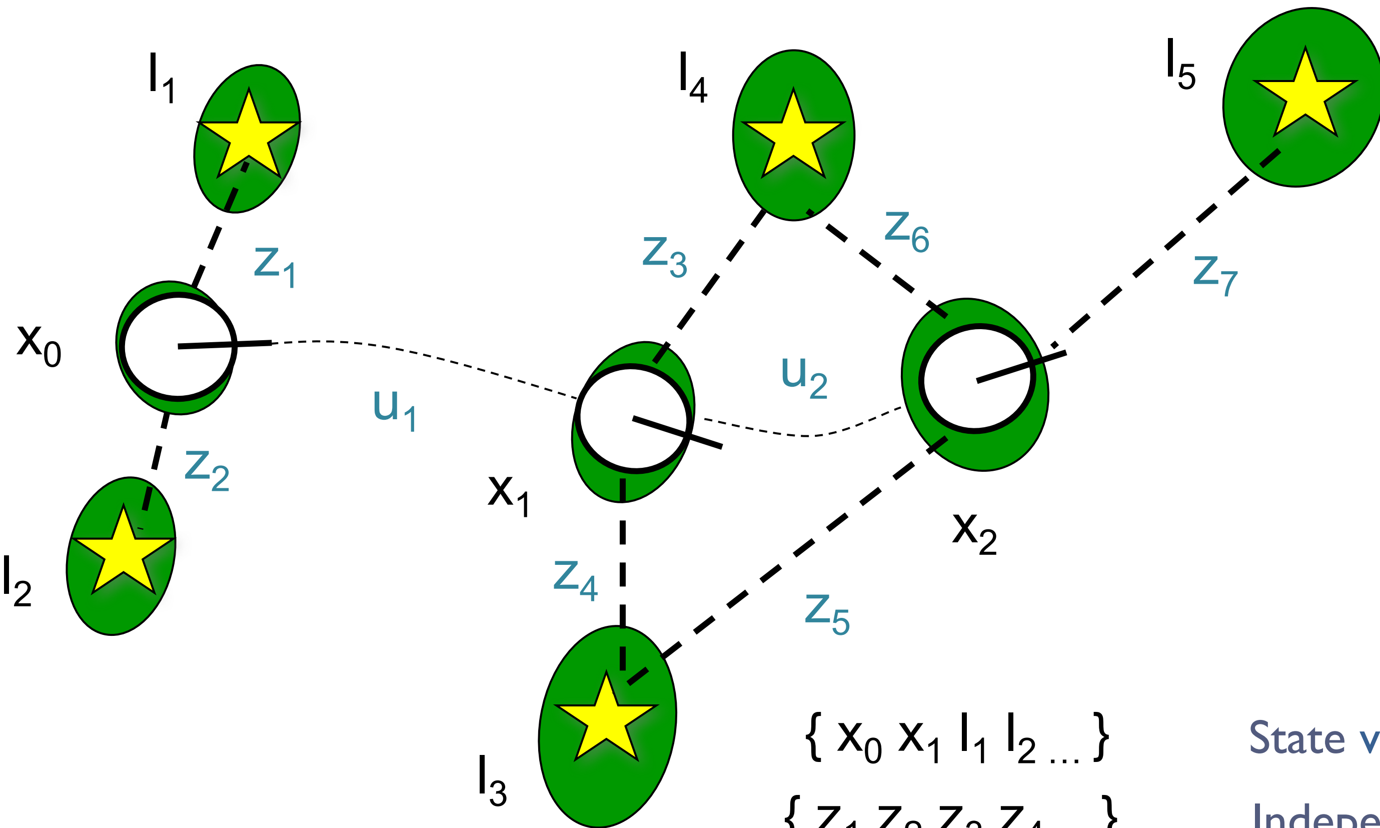
Grid Maps – Algorithm

Algorithm occupancy_grid_mapping($\{l_{t-1,i}\}, x_t, z_t$)**:**
    for all cells $\mathbf{m}_i$ do
        if $\mathbf{m}_i$ in perceptual field of z_t then
            $l_{t,i} = l_{t-1,i} + \textbf{inverse\_sensor\_model}(\mathbf{m}_i, x_t, z_t) - l_0$
        else
            $l_{t,i} = l_{t-1,i}$
        endif
    endfor
    return $\{l_{t,i}\}$

Courtesy of S. Thrun

Maximum Likelihood Estimation
The SLAM Example

SLAM – variables and measurements



$\{ x_0\ x_1\ l_1\ l_2 \ldots \}$ State variables

$\{ z_1\ z_2\ z_3\ z_{4\ldots} \}$ Independent data

$\{ u_1 \ldots \}$ Control data

Noisy Models

Motion model:

$$x_t = f_i(x_{t-1}, u_t) + v_t$$

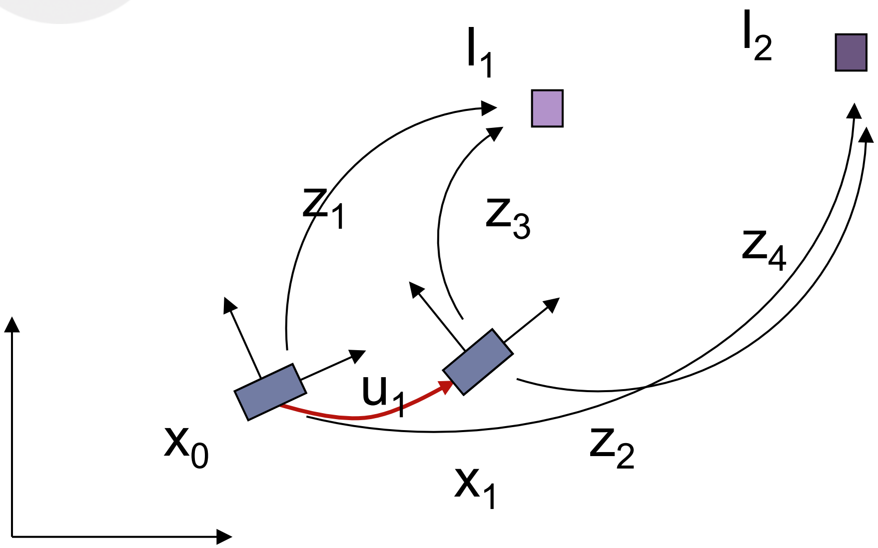
$$P(x_t \mid x_{t-1}, u_t) \propto \exp\left(-\frac{1}{2} \| f_i(x_{t-1}, u_t) - x_t \|^2_{\Sigma_{x_t}}\right)$$

Observation model:

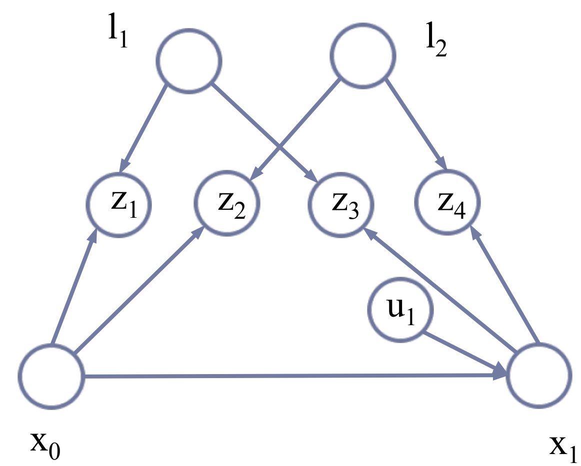
$$z_t^j = h_k(x_t, l_j) + v_n$$

$$P(z_t^j \mid x_t, l_j) \propto \exp\left(-\frac{1}{2} \| h(\mu_{x_t}, \mu_{l_j}) - z_t^j \|^2_{\Sigma_{z_t^j}}\right)$$

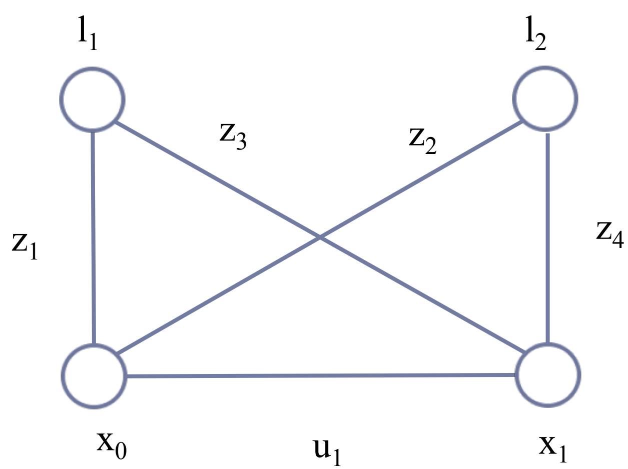
Graphical Models for SLAM



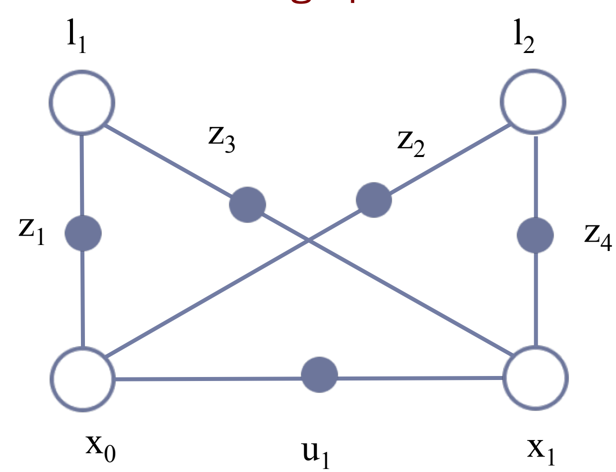
Bayesian belief network



Markov Networks



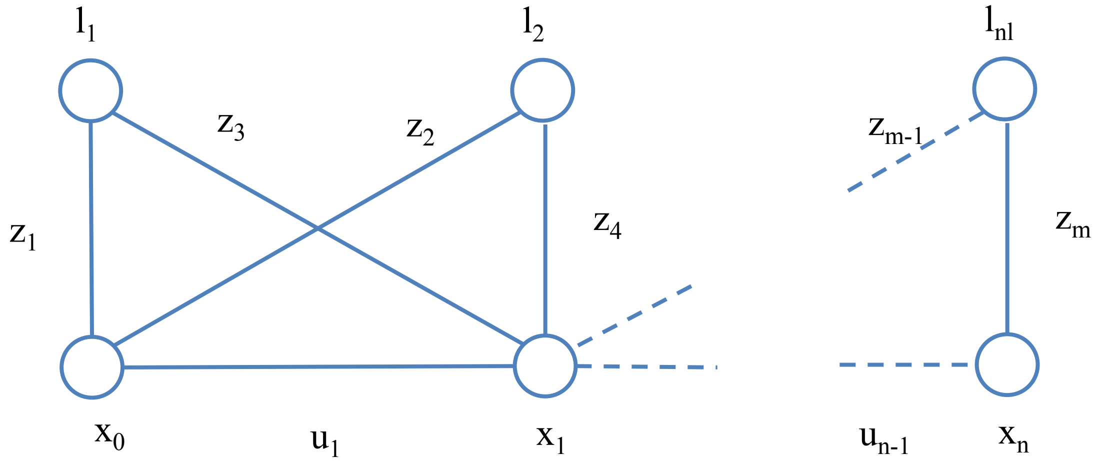
Factor graphs



Maximum Likelihood Estimation

Maximum A Posteriori estimate (MAP)

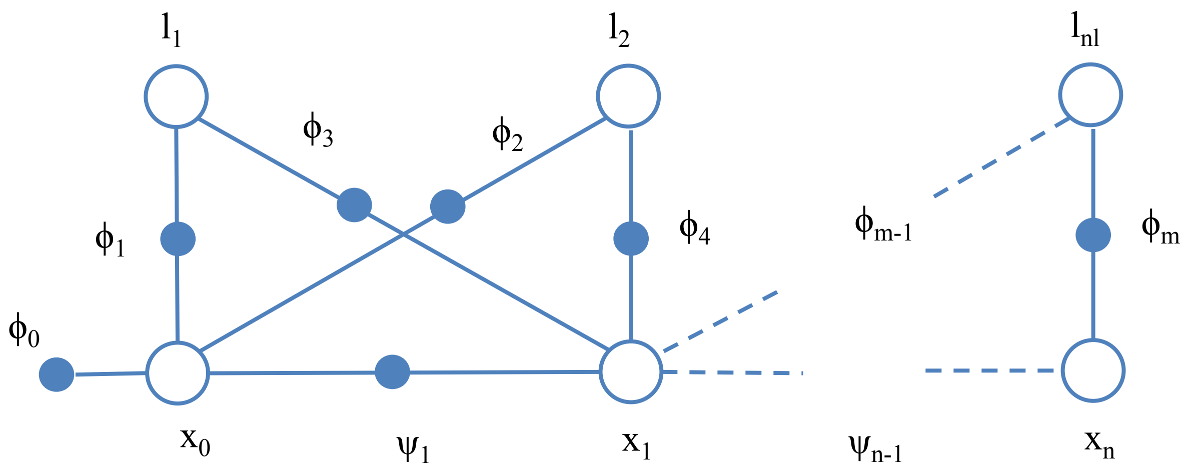
$$P(X,L) = P(\mathbf{x}_0)\prod_i^n P(x_i \mid x_{i-1}, u_i)\prod_k^m P(z_k \mid x_{i_k}, l_{j_k})$$



The configuration that maximizes the joint probability distribution

Maximum Likelihood Estimation

$$P(X, L) = \phi(\mathbf{x}_0) \prod_{i}^{n} \psi(x_{i-1}, u_i) \prod_{k}^{m} \phi(x_{i_k}, l_{j_k})$$



Factor graph expression of the joint probability distribution

Maximum Likelihood Estimation

$$P(X, L) = P(\mathbf{x}_0) \prod_i^n P(x_i \mid x_{i-1}, u_i) \prod_k^m P(z_k \mid x_{i_k}, l_{j_k})$$

Replace the multivariate normal distributions

$$\max\{P(X, L)\} = \max \left\{ \prod_k^m \exp\left(-\frac{1}{2} \| h(x_{i_k}, l_{j_k}) - z_k \|^2_{\Sigma_z} \right) \prod_i^n \exp\left(-\frac{1}{2} \| f(x_{i-1}, u_i) - x_i \|^2_{\Sigma_u} \right) \right\}$$

NIGHTMARE!!!

- log(x)

$$\operatorname{argmax}\left\{-\log\left(\prod_k^m \exp(r_k)\right)\right\} = \operatorname{argmin}\left\{\sum_k^m r_k\right\}$$

Makes everything easier!

$$\{L^*, X^*\} = \min\left\{\frac{1}{2}\sum_{k=1}^{m} \underbrace{\|h(x_{i_k}, l_{j_k}) - z_k\|}_{\text{errors}}{}_{\Sigma_z}^2 + \sum_{i=1}^{n}\frac{1}{2}\underbrace{\|f(x_{i-1}, u_i) - x_i\|}_{\text{errors}}{}_{\Sigma_u}^2\right\}$$

Nonlinear Least Squares Problem

Nonlinear Least Squares

A standard nonlinear least squares

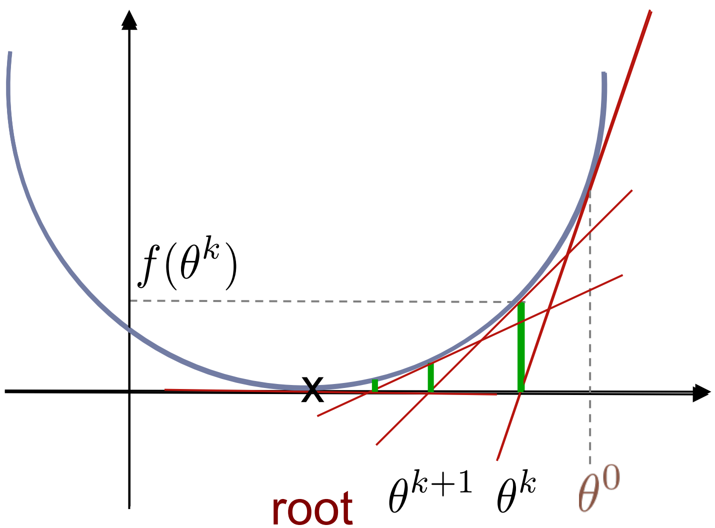
$$\boldsymbol{\theta} = \{L, X\}$$

stationary point $$\boldsymbol{\theta}^* \quad = \quad \min \{\ F(\boldsymbol{\theta})\ \}$$

$$F(\boldsymbol{\theta}) \quad = \quad \frac{1}{2} \sum_{k=1}^{m} \left\| \mathbf{r}_k(\boldsymbol{\theta}) \right\|^2$$

Newton Method

Newton methods can be used to find the root of a function.



- Start with an initial estimate: θ^0
- Calculate the tangent in this point:

$$t(\theta) = f'(\theta^k)(\theta - \theta^k) + f(\theta^k)$$

- Find the intercept:

$$t(\theta^{k+1}) = 0$$

- Iterate:

$$\theta^{k+1} = \theta^k - \frac{f(\theta^k)}{f'(\theta^k)}$$

Newton Method in Optimization

For minimizing a nonlinear function, one applies Newton method to the **first derivative**.

$$f'(\theta^*) = 0$$

stationary point

$$f'(\theta^k) \propto f'(\theta^k) + f''(\theta^k)\,\Delta\theta = 0$$

$$\Delta\theta = \theta - \theta^k$$

$$\theta^{k+1} = \theta^k - \frac{f'(\theta^k)}{f''(\theta^k)}$$

Needs the second derivative

Nonlinear Least Squares

$$F(\boldsymbol{\theta}) = \frac{1}{2}\sum_{k=1}^{m} \|\mathbf{r}_k(\boldsymbol{\theta})\|^2 \qquad \boldsymbol{\theta}^* = \min\{\, F(\boldsymbol{\theta}) \,\}$$

Nonlinear residuals: $\mathbf{r}(\boldsymbol{\theta}) = [r_1, \ldots, r_m]^\top$

Linearize: $\tilde{\mathbf{r}}(\boldsymbol{\theta}) = \mathbf{r}(\boldsymbol{\theta}^0) + J(\boldsymbol{\theta}^0)\underbrace{(\boldsymbol{\theta} - \boldsymbol{\theta}^0)}_{\text{correction } \boldsymbol{\delta}}$

Linear Least Squares:

$$\frac{1}{2}\sum_{k=1}^{m} \|r_{0_k} + J_k\delta_k\|^2 = \frac{1}{2}\|\mathbf{r_0}\|^2 + \boldsymbol{\delta}^\top J^\top \mathbf{r_0} + \frac{1}{2}\boldsymbol{\delta}^\top J^\top J\boldsymbol{\delta}$$

Linear Least Squares

We need to find the minimum of :

$$L(\boldsymbol{\delta}) \quad = \quad \frac{1}{2}\|\mathbf{r_0}\|^2 + \boldsymbol{\delta}^\top J^\top \mathbf{r_0} + \frac{1}{2}\boldsymbol{\delta}^\top J^\top J\boldsymbol{\delta}$$

1st derivative:

$$L(\boldsymbol{\delta})' \quad = \quad J^\top \mathbf{r_0} + J^\top J\boldsymbol{\delta}$$

The minimum is where the 1st derivative cancels

$$J^\top \mathbf{r_0} + J^\top J\boldsymbol{\delta} = 0$$

Correction:

$$\boldsymbol{\delta}^*$$

Jacobians and "Hessians"

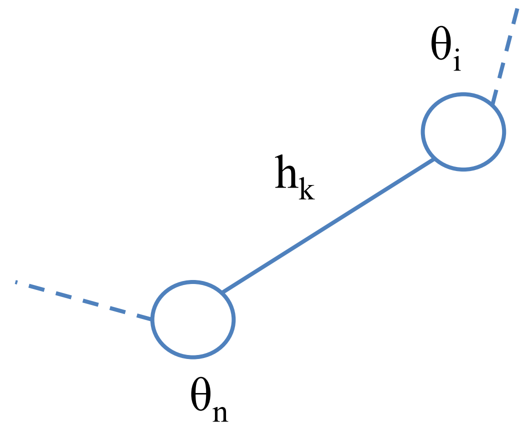
Jacobian — Hessian (approx.)

$$L(\boldsymbol{\delta})' = \underbrace{J^\top}_{\text{Jacobian}} \mathbf{r_0} + \underbrace{J^\top J}_{\text{Hessian (approx.)}} \boldsymbol{\delta}$$

$$J_k = \begin{bmatrix} \frac{\delta r_k}{\delta \theta_1} \\ \frac{\delta r_k}{\delta \theta_2} \\ \vdots \\ \frac{\delta r_k}{\delta \theta_n} \end{bmatrix} \qquad H_k = \begin{bmatrix} \frac{\delta^2 r_k}{\delta \theta_1 \delta \theta_1} & \frac{\delta^2 r_k}{\delta \theta_1 \delta \theta_2} & \cdots & \frac{\delta^2 r_k}{\delta \theta_1 \delta \theta_n} \\ \frac{\delta^2 r_k}{\delta \theta_2 \delta \theta_1} & \frac{\delta^2 r_k}{\delta \theta_2 \delta \theta_2} & \cdots & \frac{\delta^2 r_k}{\delta \theta_2 \delta \theta_n} \\ \vdots & & & \\ \frac{\delta^2 r_k}{\delta \theta_n \delta \theta_1} & \frac{\delta^2 r_k}{\delta \theta_n \delta \theta_2} & \cdots & \frac{\delta^2 r_k}{\delta \theta_n \delta \theta_n} \end{bmatrix}$$

Jacobians and "Hessians"

Each measurement affects few variables (2 in general):



$$J_k = \begin{bmatrix} 0 \\ \vdots \\ \frac{\delta r_k}{\delta \theta_i} \\ 0 \\ \vdots \\ \frac{\delta r_k}{\delta \theta_n} \end{bmatrix} \qquad H_k = \begin{bmatrix} 0 & \dots & 0 & \dots & 0 \\ 0 & \dots & \frac{\delta^2 r_k}{\delta \theta_i \delta \theta_i} & 0 \dots 0 & \frac{\delta^2 r_k}{\delta \theta_i \delta \theta_n} \\ \vdots & & \vdots & & \vdots \\ 0 & \dots & \frac{\delta^2 r_k}{\delta \theta_n \delta \theta_i} & 0 \dots 0 & \frac{\delta^2 r_k}{\delta \theta_n \delta \theta_n} \end{bmatrix}$$

Gauss-Newton

$$F(\boldsymbol{\theta}) = \frac{1}{2}\sum_{k=1}^{m} \|\mathbf{r}_k(\boldsymbol{\theta})\|^2$$

while 1

linearize $F(\theta)$ in θ^i ➔ $L(\delta)$

solve $L(\delta)' = 0$ obtain δ^*

if $\text{norm}(\delta^*) < \text{threshold}$

done

update $\theta^{i+1} = \theta^i + \delta^*$

SLAM - Solve

$$L(\boldsymbol{\delta}) \quad = \quad \|\mathbf{b}\|^2 + \boldsymbol{\delta}^\top A^\top \mathbf{b} + \frac{1}{2}\boldsymbol{\delta}^\top A^\top A\boldsymbol{\delta}$$

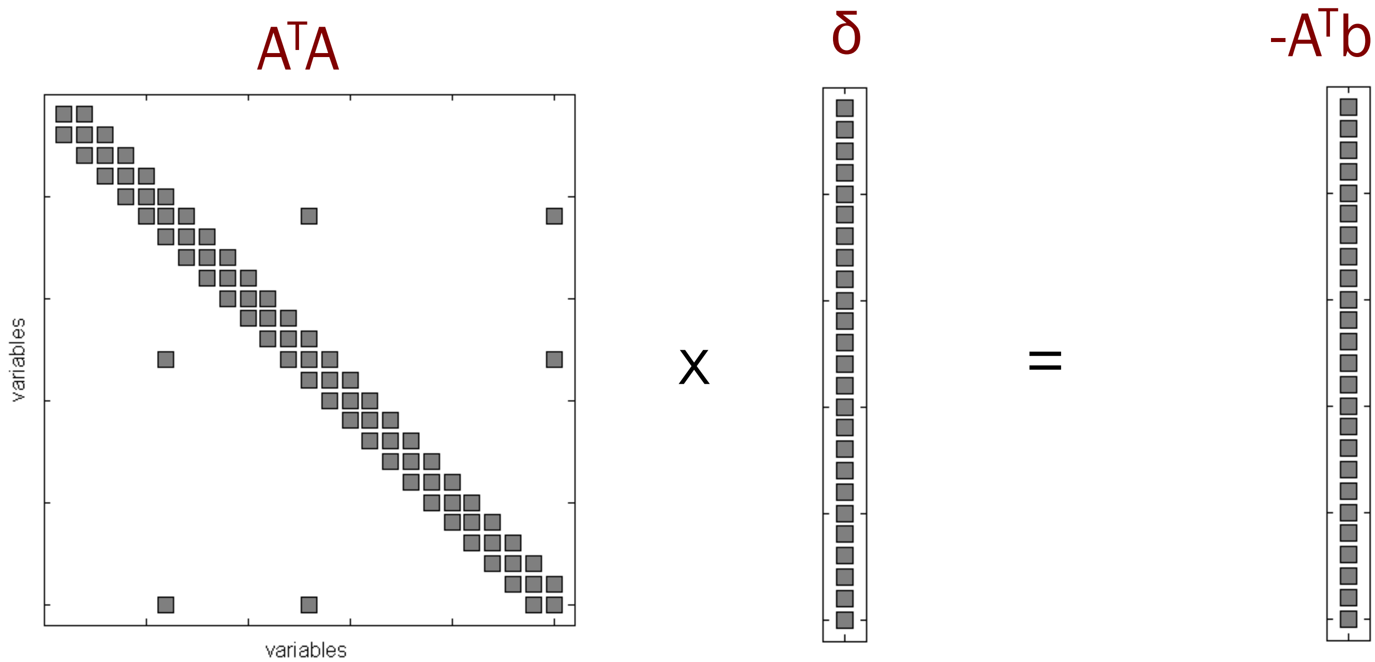
The min is where the first derivative cancels!

$$L(\boldsymbol{\delta})' \quad = \quad A^\top \mathbf{b} + A^\top A\boldsymbol{\delta} = 0$$

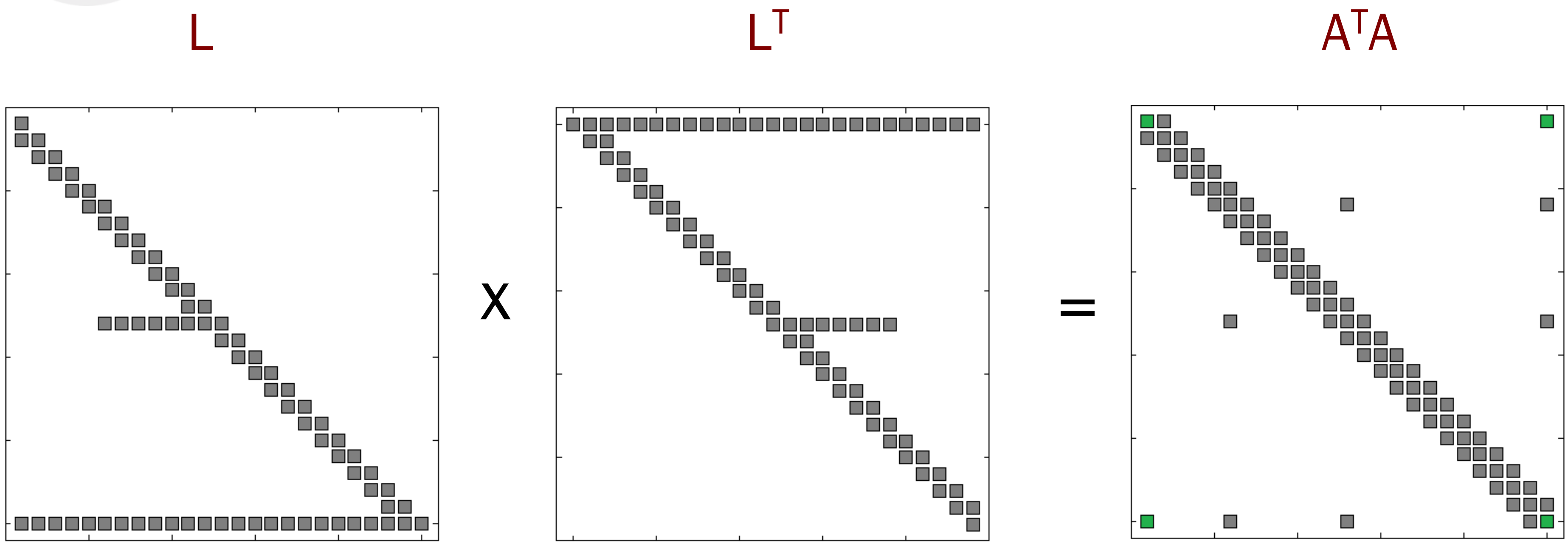
$$A^\top A\boldsymbol{\delta} = -A^\top \mathbf{b}$$

Normal Equation

$$A^\top A\boldsymbol{\delta} = -A^\top \mathbf{b}$$

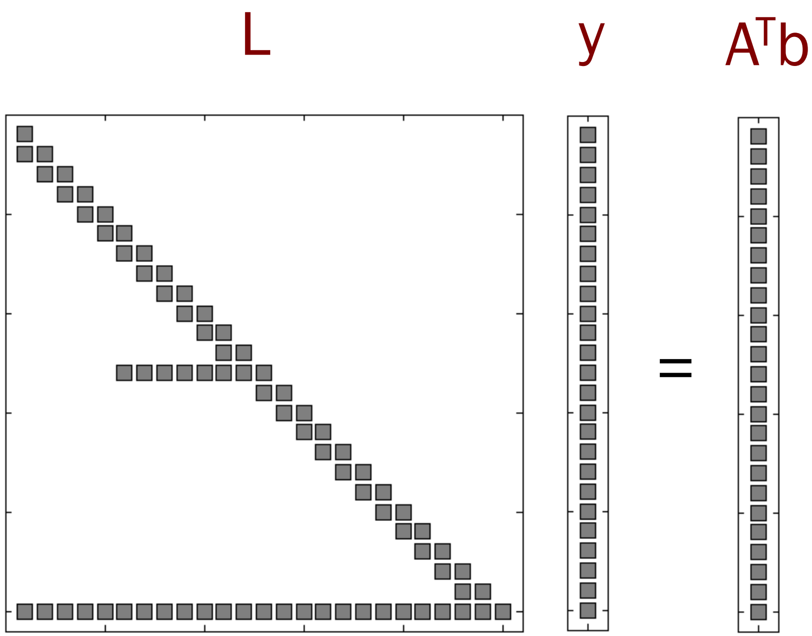


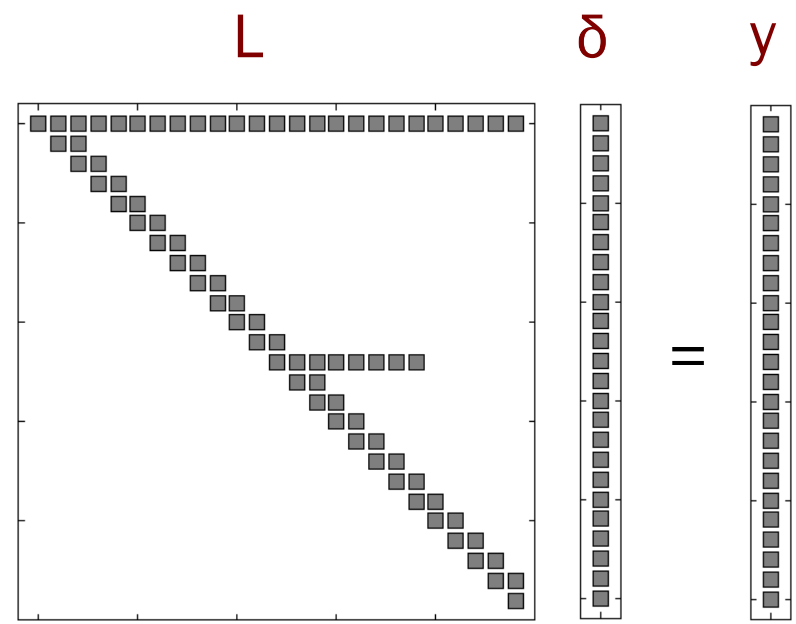
Matrix Factorization



- Symmetric positive definite matrix A^TA has Cholesky factorization $A^TA = LL^T$ where L is lower triangular matrix with positive diagonal entries.

Matrix Factorization

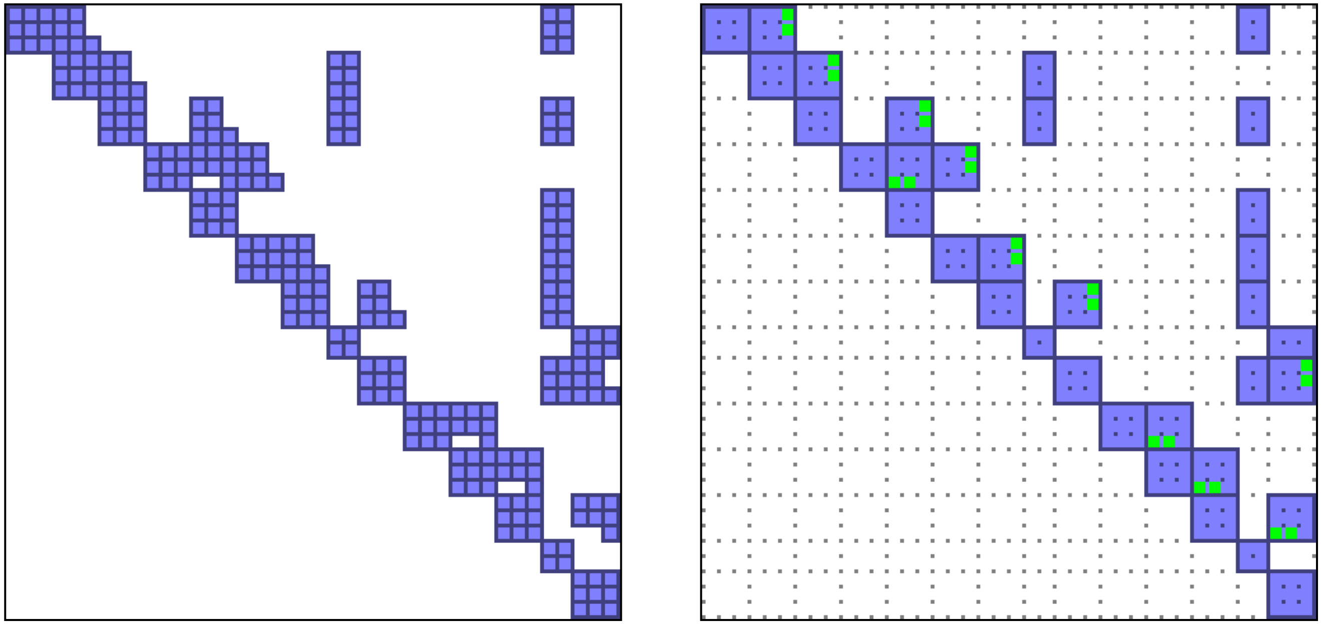




- Linear system $A^TA\ \delta = A^Tb$ can then be solved by **forward substitution** in lower triangular system $Ly = A^Tb$, followed by **back-substitution** in upper triangular system $L^T\ \delta = y$

Sparse Matrices

- A matrix is called **sparse** if many of its entries are zero



- A **block matrix** is a matrix which is interpreted as partitioned into sections called blocks that can be manipulated at once

Sparse Algebra

SuiteSparse

v4.5.3: Sparse QR and Cholesky on the GPU

http://faculty.cse.tamu.edu/davis/suitesparse.html

SLAM++

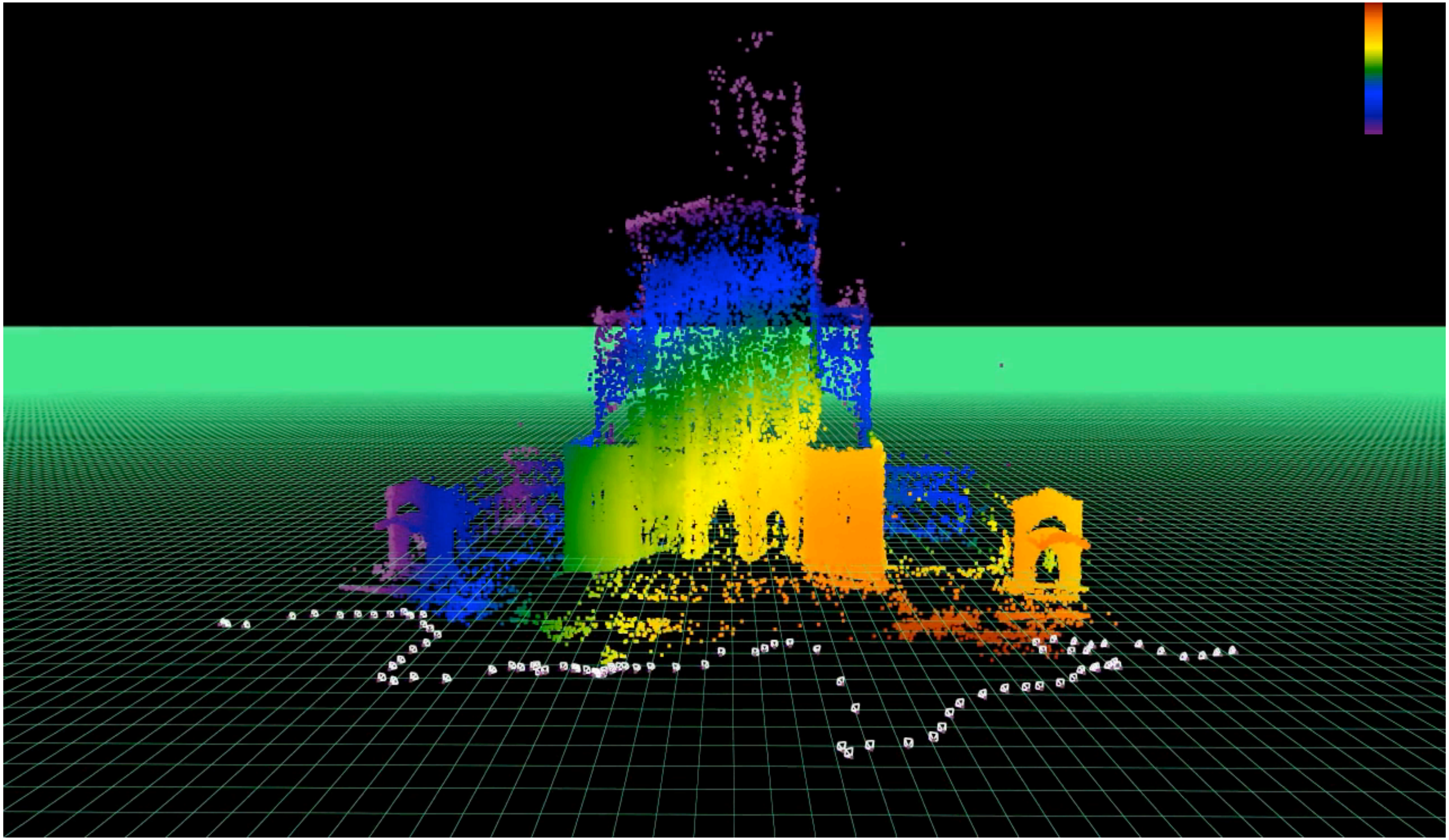
high-performance nonlinear least squares solver for graph problems

Brought to you by: iviorela, swajnautcz

http://sourceforge.net/projects/slam-plus-plus/

Pose - SLAM

Structure From Motion



AUSTRALIAN CENTRE FOR

ROBOTIC VISION

Australian Government

Australian Research Council

QUT

THE UNIVERSITY of ADELAIDE

THE AUSTRALIAN NATIONAL UNIVERSITY

MONASH University

CSIRO

UNIVERSITY OF OXFORD

Inria INVENTORS FOR THE DIGITAL WORLD

Imperial College London

ETH zürich

Georgia Tech