Least Squares and SLAM
Landmark-SLAM

Giorgio Grisetti

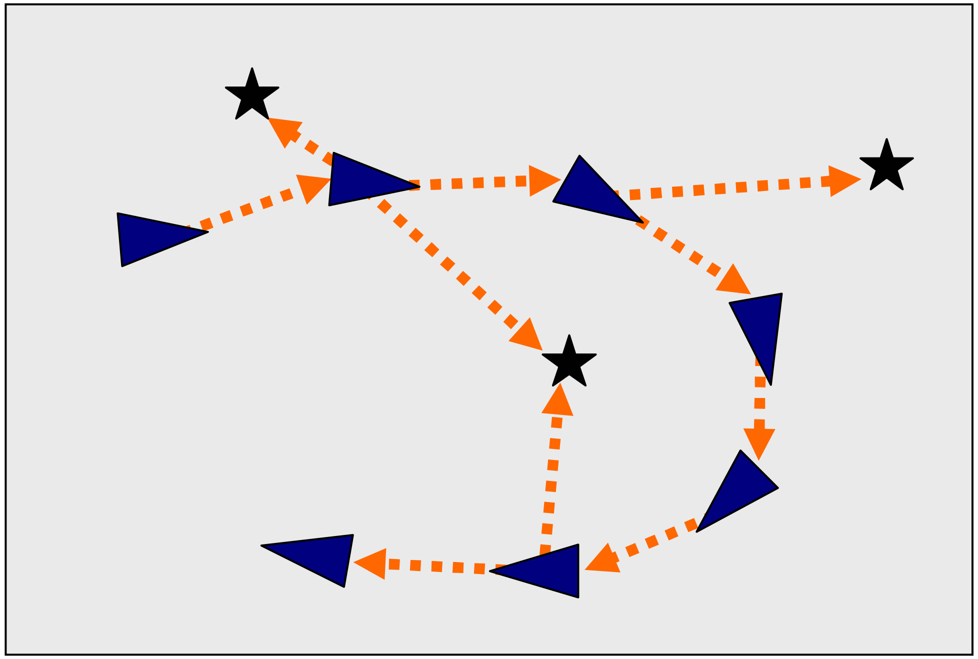
Part of the material of this course is taken from the Robotics 2 lectures given by G.Grisetti, W.Burgard, C.Stachniss, K.Arras, D. Tipaldi and M.Bennewitz

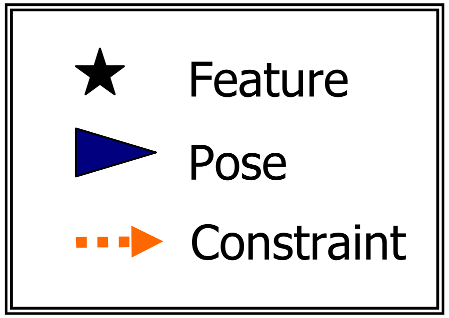
The Graph

- The SLAM graph we have seen so far consists of:
  - Vertices, representing robot poses $\boldsymbol{x_i} =(x,y,\theta)^T$
  - Edges, representing virtual observations between robot poses $\boldsymbol{z_{ij}}$ = < $(x,y,\theta)^T{}_{ij}$, Ω_{ij} >.
- In this lecture we will:
  - Extend the system to operate on an extended graph (i.e. with landmarks)
  - See how to obtain a pose-graph out of a landmark based system.

The Graph in the General Case

- A node represents a state variable:
  - A robot position, or
  - A landmark in the environment
- An edge represents a measurement:
  - Landmark observation
  - Odometry measurement
- The minimization seeks for the configuration of landmarks and robot poses that is most consistent with the observations in the edges.





Landmarks in x-y

- A landmark represents a 2D point in the world $\boldsymbol{x_j} = (x\ y)^T$.
- The robot observes the landmark relative to its current location.
- Synthetic measurement

Landmarks in x-y

- A landmark represents a 2D point in the world $\boldsymbol{x_j} = (x\ y)^T$.
- The robot observes the landmark relative to its current location.
- Synthetic measurement

$$\widehat{\mathbf{z}}_{ij}(\mathbf{x}_i, \mathbf{x}_j) \quad = \quad \mathbf{R}_i^T(\mathbf{x}_j - \mathbf{t}_i)$$

Robot

Landmark

Robot translation

- Error Function

Landmarks in x-y

- A landmark represents a 2D point in the world $\mathbf{x_j} = (x\ y)^T$.
- The robot observes the landmark relative to its current location.
- Synthetic measurement

$$\hat{\mathbf{z}}_{ij}(\underbrace{\mathbf{x}_i}_{\text{Robot}}, \underbrace{\mathbf{x}_j}_{\text{Landmark}}) \quad = \quad \mathbf{R}_i^T(\mathbf{x}_j - \underbrace{\mathbf{t}_i}_{\text{Robot translation}})$$

- Error Function

$$\begin{aligned} \mathbf{e}_{ij}(\mathbf{x}_i, \mathbf{x}_j) &= \hat{\mathbf{z}}_{ij} - \mathbf{z}_{ij} \\ &= \mathbf{R}_i^T(\mathbf{x}_j - \mathbf{t}_i) - \mathbf{z}_{ij} \end{aligned}$$

Landmarks in the Bearing only Case

- A landmark still represents a 2D point, but we can observe only the bearing.
- Synthetic Measurement
- Error function:

Landmarks in the Bearing only Case

- A landmark still represents a 2D point, but we can observe only the bearing.
- Synthetic Measurement

$$\hat{\mathbf{z}}_{ij}(\mathbf{x}_i, \mathbf{x}_j) \quad = \quad \mathrm{atan}\frac{(\mathbf{x}_j - \mathbf{t}_i).y}{(\mathbf{x}_j - \mathbf{t}_i).x} - \theta_i$$

Robot

Landmark

Robot-landmark angle

Robot orientation

- Error function:

Landmarks in the Bearing only Case

- A landmark still represents a 2D point, but we can observe only the bearing.
- Synthetic Measurement

$$\hat{\mathbf{z}}_{ij}(\mathbf{x}_i, \mathbf{x}_j) \quad = \quad \mathrm{atan}\frac{(\mathbf{x}_j - \mathbf{t}_i).y}{(\mathbf{x}_j - \mathbf{t}_i).x} - \theta_i$$

Robot

Landmark

Robot-landmark angle

Robot orientation

- Error function:

Landmark Bearing

$$\mathbf{e}_{ij}(\mathbf{x}_i, \mathbf{x}_j) \quad = \quad \mathrm{atan}\frac{(\mathbf{x}_j - \mathbf{t}_i).y}{(\mathbf{x}_j - \mathbf{t}_i).x} - \theta_i - \mathbf{z}_j$$

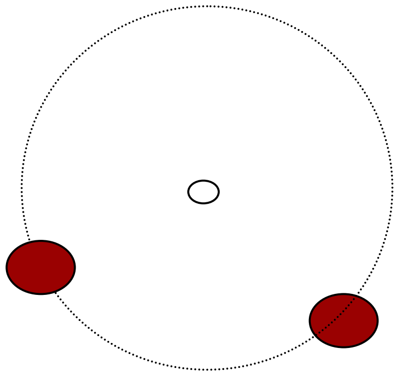
Considerations about the Rank of the Hessian

- What is the rank of the Hessian of a 2D landmark-pose constraint?
  - The jacobian is a 2x3 matrix
  - The Hessian cannot be more than 2

- What is the rank of the hessian for a bearing-only constraint?
  - The Jacobian is a 1x3 matrix -> rank=1

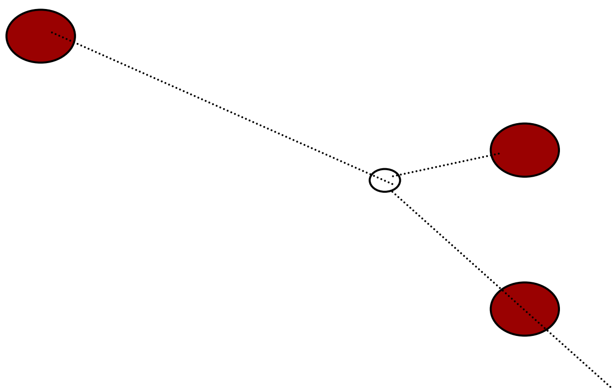
Again on the Rank

- If I see 1 landmark (x-y) where can the robot be?



It can lie on a circle inf^1 solutions

- If I observe the bearing of 1 landmark where can the robot be?



It can be everywhere on the plane. Constraint on orientation: inf^2.

Rank

- In the landmark case the system can be under-determined.
- The rank of the Hessian is **at most** equal to the sum of the ranks of the constraints.
- Now, looking at the rank:
  - How many bearing observations do I need to resolve for a robot pose?
  - How many 2d-landmark observations do I need to resolve for a robot pose?
- **To determine a unique solution, the system should be full rank!**

Dealing with under-determined Systems

- In the general case one cannot guarantee that the system will be over-determined.
  - Certain landmarks can be observed only once
  - The robot might have no odometry
- We can still deal with these situations by adding a “damping” factor to the Hessian.
  - Instead of solving $\boldsymbol{H}\,\boldsymbol{\Delta x} = -\boldsymbol{b}$, we solve

$$(\boldsymbol{H} + \lambda \boldsymbol{I})\,\boldsymbol{\Delta x} = -\boldsymbol{b}$$

  - The damping factor $\lambda \boldsymbol{I}$ makes the system positive definite (from semi-positive), by adding additional constraints that “drag” the increments towards 0.
  - What happens when $\lambda \gg |\boldsymbol{H}|$?

Levenberg Marquardt (simplified)

- This "damping" trick can be used to regulate the convergence by using appropriate backup/restore actions.

```
x: the initial guess
While (! converged)
    λ = λ_init
    <H,b> = buildLinearSystem(x);
    E = error (x)
    x_old = x;
    Δx  = solveSparse( (H + λ I) Δx = -b);
    x += Δx;
    If (E<error(x)){
        x  = x_old;
        λ *= 2;
    } else {
        λ */ 2;
    }
```

Fixing a (set of) variables

- Assume that the value of certain variables during the optimization is known a priori.
- We may want to optimize all others and keep these fixed.
- How?

Fixing a (set of) variables

- Assume that the value of certain variables during the optimization is known a priori.
- We may want to optimize all others and keep these fixed.
- How?
- If a variable is not optimized, it simply “disappears” from the linear system

Fixing a (set of) variables

- Assume that the value of certain variables during the optimization is known a priori.
- We may want to optimize all others and keep these fixed.
- How?
- If a variable is not optimized, it simply “disappears” from the linear system
- Construct the full system
- Suppress the rows and the columns corresponding to the variables to fix

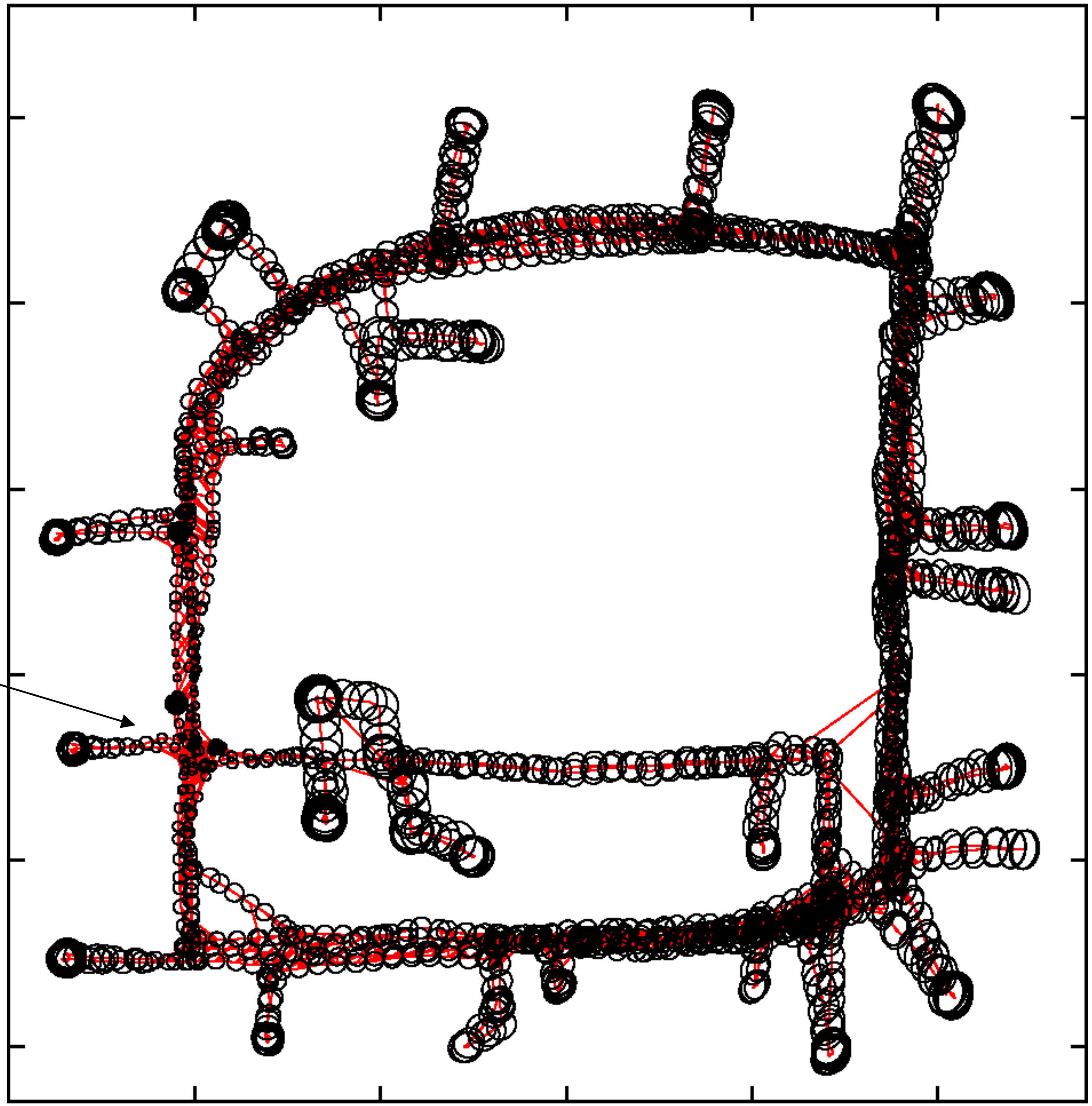
Determining the Relative Uncertainty

- The Hessian represents the inverse covariance of the likelihood around the linearization point
- Inverting the Hessian gives the covariance matrix (which is dense)
- The diagonal blocks of the covariance matrix represent the absolute uncertainties of the corresponding variables

- To determine the relative uncertainty between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$:
  - Construct the full Hessian
  - Suppress the rows and the columns o $\boldsymbol{x}_i$ (fix it)
  - Compute the *j,j* block of the inverse. This block will contain the covariance matrix of $\mathbf{x}_j$ w.r.t. $\boldsymbol{x}_i$, which has been fixed.

Example

robot



Conclusions

- We now know
  - How to incorporate landmarks in the map
  - How to determine the relative uncertainties
  - How to embed prior knowledge about the position of some parts of the map