Robótica Móvil

SLAM: Simultaneous Localization and Mapping

Taihú Pire

Laboratorio de Robótica

CONICET

UNIVERSIDAD NACIONAL DE ROSARIO
CONFINGERE HOMINEM COGITANTEM

C I F A S I S

Temario para estas slides

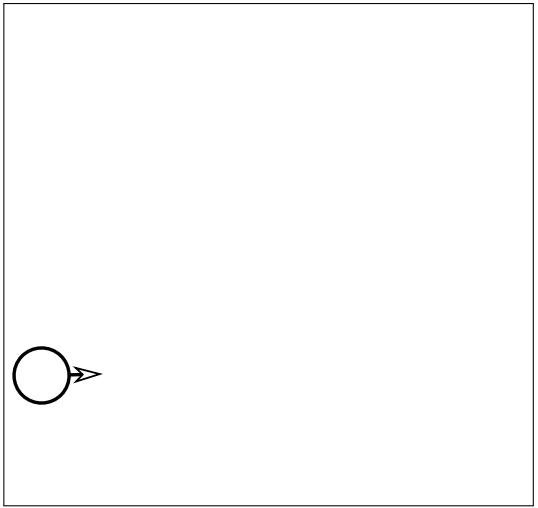
- Graph-SLAM / Factor Graph
- Loop-Closure
- Bundle Adjustment

¿Qué es SLAM?

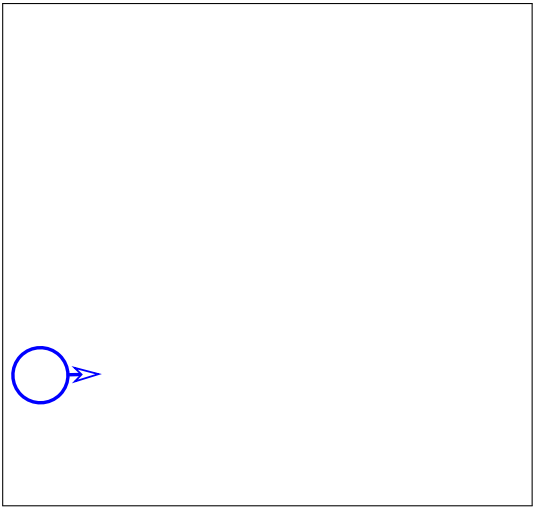
Para que un robot móvil pueda navegar de manera autónoma es necesario que conozca su ubicación y cuente con una representación del entorno donde se encuentra. Estos problemas se conocen como el problema Localización y el problema de Mapeo. En el caso más general, donde no se cuenta con un la localización del robot ni con un mapa a priori del entorno, dichas problemas se abordan de manera simultánea. Esto da origen al problema de SLAM (*Simultaneous Localizacion and Mapping*).

SLAM es el problema de resolver la localización y el mapeo al mismo tiempo.

Ejemplo de SLAM



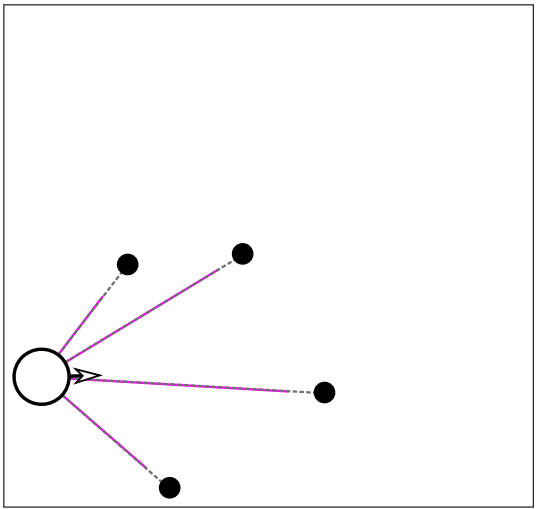
Realidad



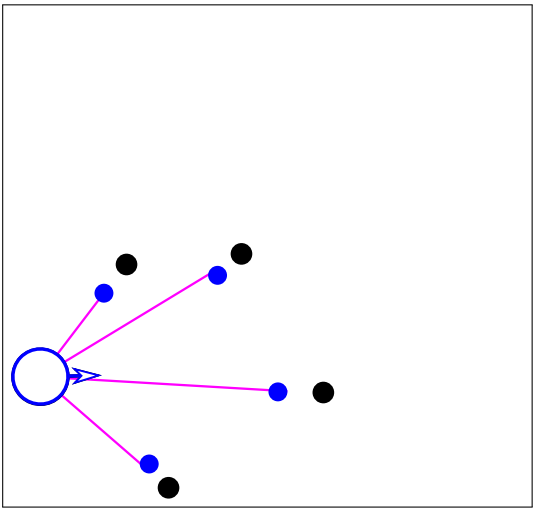
Sistema de SLAM del robot

Ejemplo de SLAM

mejorar haciendo que los map points también se ajusten

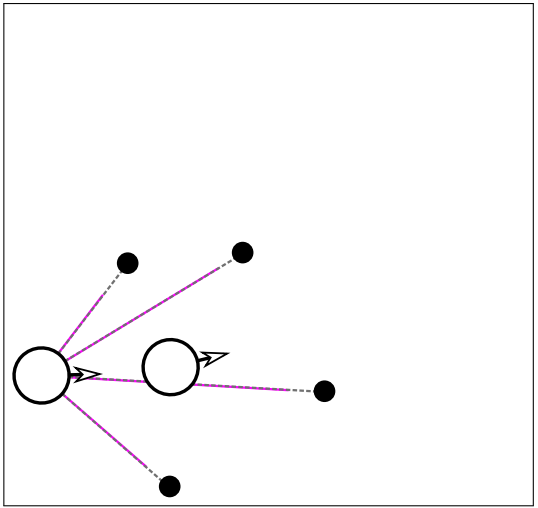


Realidad

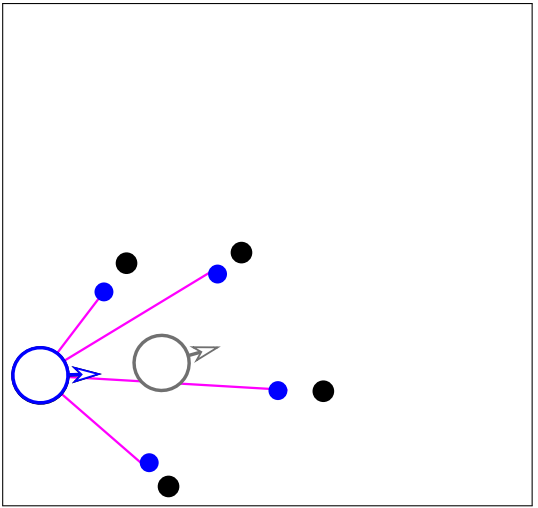


Sistema de SLAM del robot

Ejemplo de SLAM

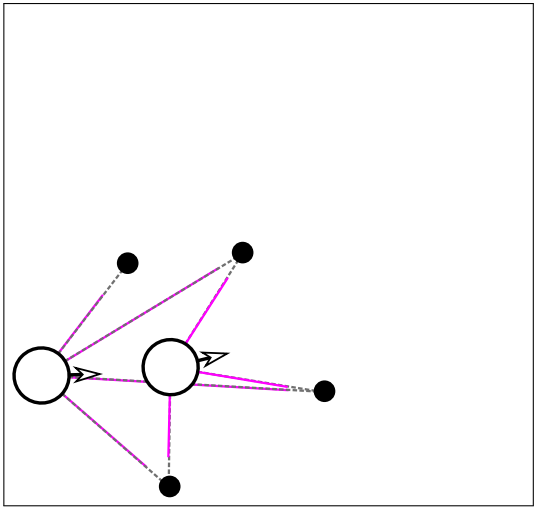


Realidad

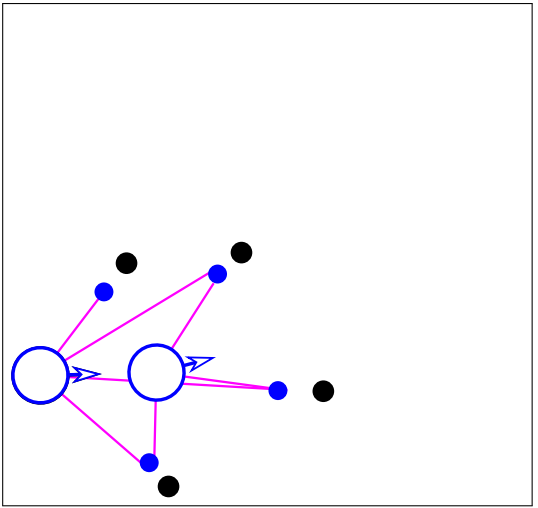


Sistema de SLAM del robot

Ejemplo de SLAM

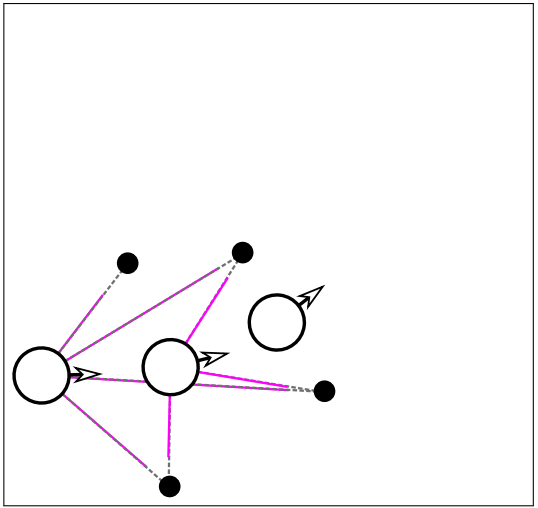


Realidad

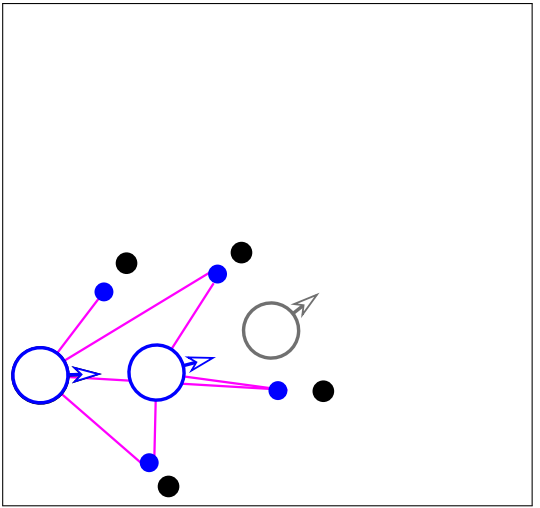


Sistema de SLAM del robot

Ejemplo de SLAM

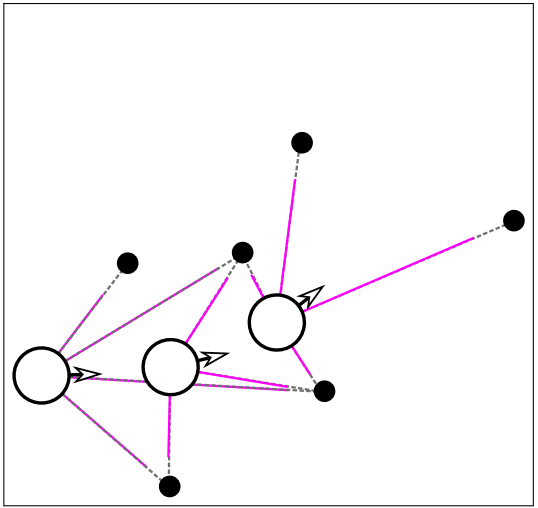


Realidad

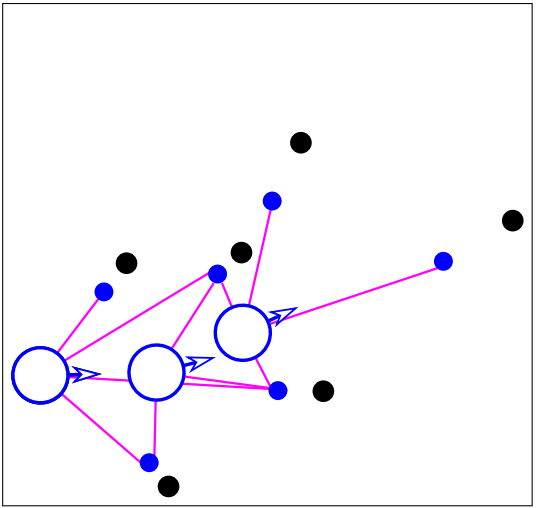


Sistema de SLAM del robot

Ejemplo de SLAM

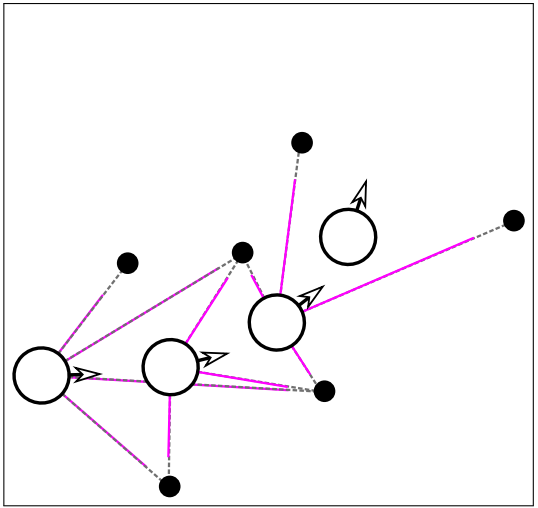


Realidad

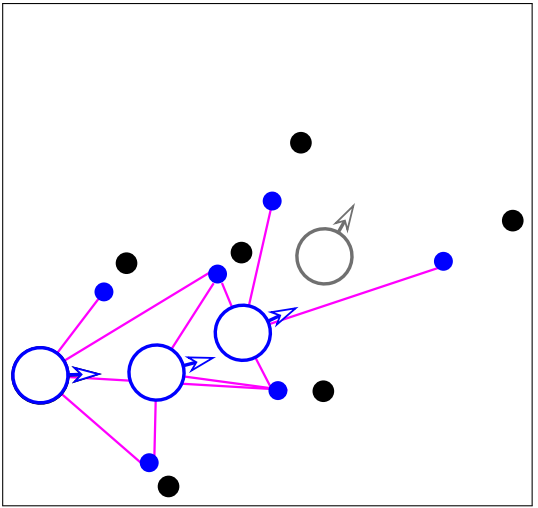


Sistema de SLAM del robot

Ejemplo de SLAM

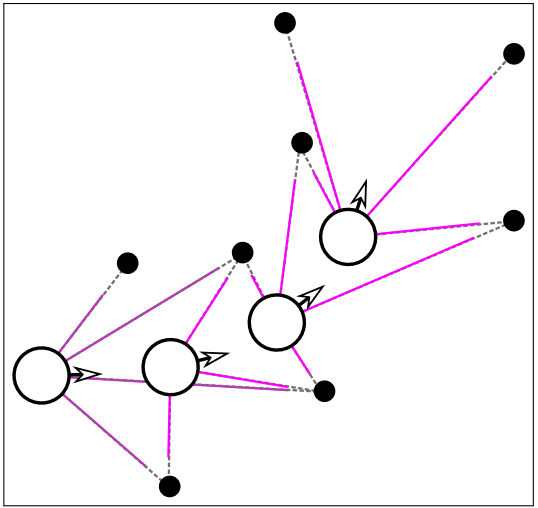


Realidad

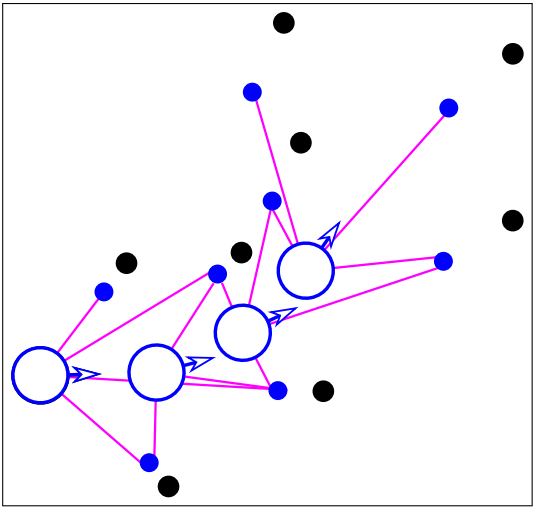


Sistema de SLAM del robot

Ejemplo de SLAM



Realidad



Sistema de SLAM del robot

Aquitectura general de SLAM

Sensor data

Front-end

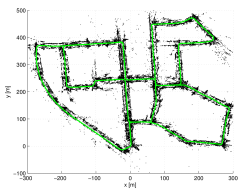
feature extraction

data association:
- short-term (feature tracking)
- long-term (loop closure)

Back-end

MAP estimation

SLAM estimate

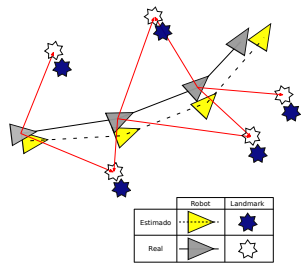


Tipos de SLAM Back-ends

- Basados en Kalman Filter (EKF-SLAM)
- Basados en Particle Filter (FastSLAM, Rao-Blackwellized Particle Filter, Gmapping)
- Basados en Least-Squares (Graph-SLAM, Bundle Adjustment)
  - Pose-Graph (solo contiene las poses del robot, el mapa es marginalizado[1])
  - Factor-Graph (contiene poses y landmarks)

Herramientas de optimización

- Ceres
- GTSAM
- g2o

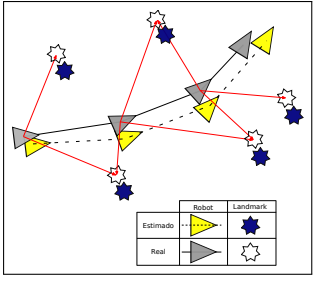


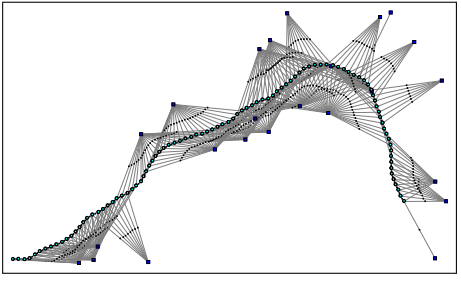
[1]Marginalizar es el proceso de remover variables sin perder información.

Graph-SLAM

Graph-SLAM: construir un grafo y encontrar una configuración de nodos que minimiza el error introducido por las restricciones (aristas)

- Se utiliza un grafo para representar el problema.
- Los nodos representan poses o ubicaciones de landmarks.
- Las aristas son observaciones de landmarks o mediciones de odometría
- La minimización optimiza las poses del robot y la ubicación del los landmarks
- Observar áreas previamente vistas genera restricciones en el grafo



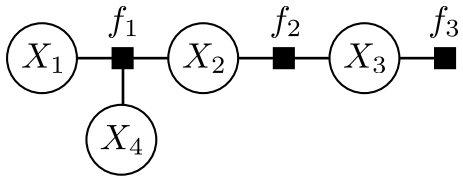


Factor-Graph

Factor-Graph

Un Factor-graph es un término matemático, un grafo bipartito representando la factorización de una función. Esto significa que podemos tomar una función por ejemplo $g(.)$ y descomponerla mediante el producto de funciones $f(.)$,

$$g(X_1, \ldots, X_n) = \prod_i f_i(S_i) \quad \text{con} \quad S_i \subseteq \{X_1, \ldots, X_n\}$$



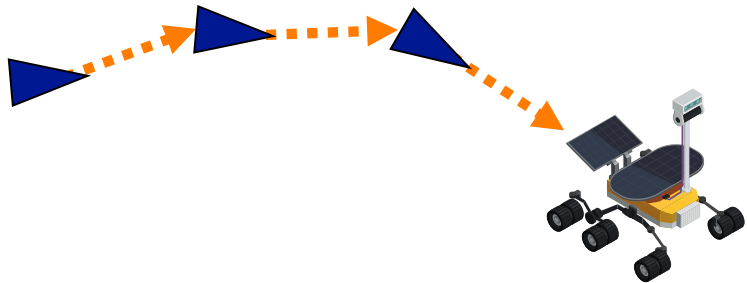
$$g(X_1, X_2, X_3, X_4) = f_1(X_1, X_2, X_4) f_2(X_2, X_3) f_3(X_3)$$

Factor-Graph

- Los Factor-Graph nos permiten representar una distribución de probabilidad conjunta (distribución que gobierna sobre todas las variables) como un producto de probabilidades más chicas (que dependen de un menor número de variables).
- Al igual que las redes de Bayes o redes de Markov, podemos utilizar los Factor-Graph para describir cómo las variables dependen entre si. Y podemos correr diferentes algoritmos sobre estos Factor-Graph para inferir información de una manera eficiente.
- Ejemplo de algorimo que trabaja sobre Factor-graph, es el Algoritmo de Sum-product para el computo de distribuciones marginales (distribuciones que solamente dependen un subconjunto de variables).
- En el contexto de robótica los Factor-graph son utilizados para especificar problemas de mínimos cuadrados. El factor graph nos permite representar cómo ciertos estados dependen o estan relacionados entre sí basados en la información que tenemos de las mediciones de los sensores (almacenada en los factores).

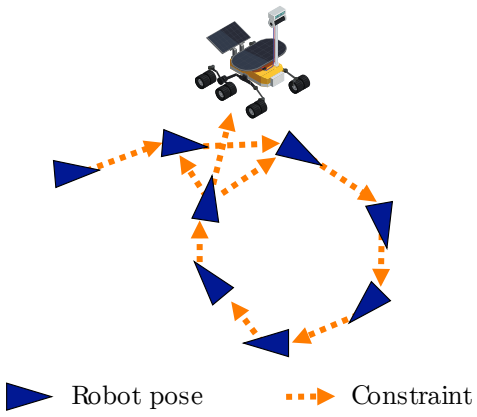
Graph-based SLAM usando Pose-Graph

- Las restricciones conectan las poses del robot mientras se desplaza
- Las restricciones tiene ruido

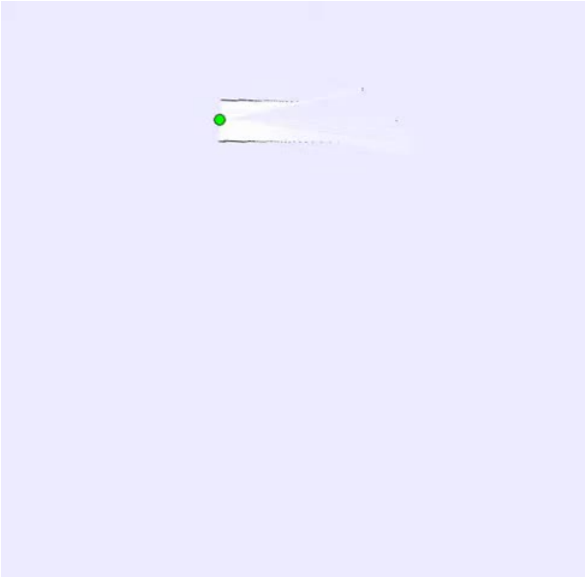


Graph-based SLAM usando Pose-Graph

- Observando áreas previamente vistas se generan restricciones entre poses no consecutivas (*Loop Closure*)

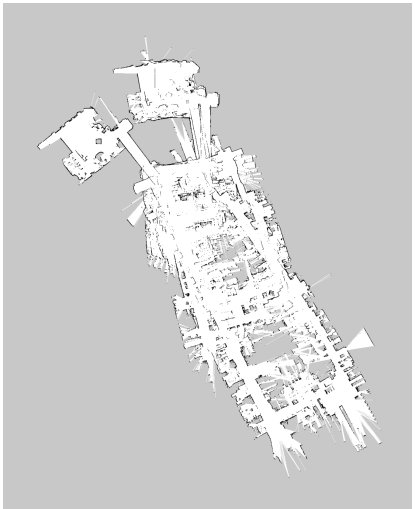


2D Pose-Graph con LiDAR



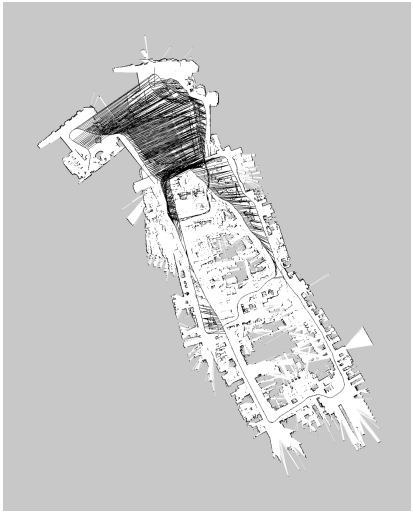
Graph-based SLAM usando Pose-Graph

- Cada nodo es una pose del robot junto con su medición de laser (no hay landmarks)
- Cada arista corresponde a una restricción espacial entre los nodos que relaciona.



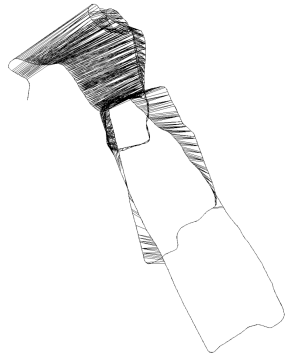
Graph-based SLAM usando Pose-Graph

- Cada nodo es una pose del robot junto con su medición de laser (no hay landmarks)
- Cada arista corresponde a una restricción espacial entre los nodos que relaciona.



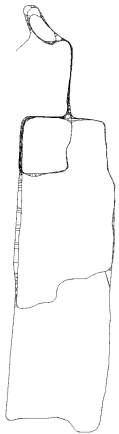
Graph-based SLAM usando Pose-Graph

- Una vez que tenemos el grafo, obtenemos el mapa mas probable mediante la corrección de los nodos



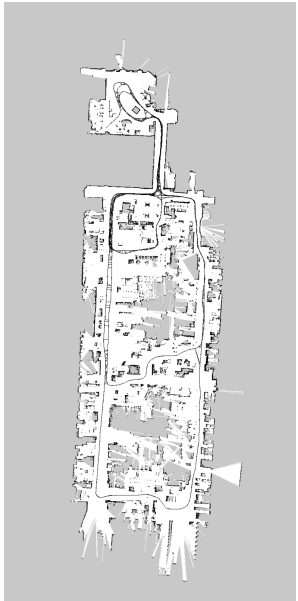
Graph-based SLAM usando Pose-Graph

- Una vez que tenemos el grafo, obtenemos el mapa mas probable mediante la corrección de los nodos
- así...



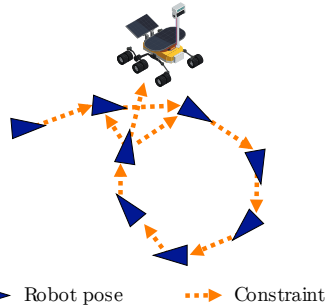
Graph-based SLAM usando Pose-Graph

- Una vez que tenemos el grafo, obtenemos el mapa mas probable mediante la corrección de los nodos
- así...
- Dibujamos el mapa basandonos en las poses corregidas



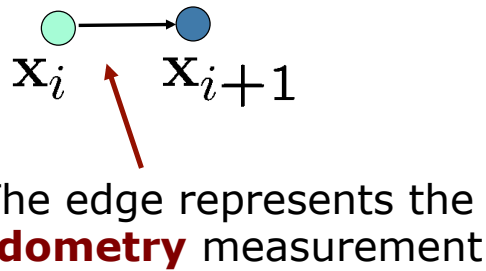
El grafo en pose-graph

- Consiste en n nodos $\mathbf{x} = \mathbf{x}_{1:n}$
- Cada $\mathbf{x}_i$ es una pose del robot en el tiempo t_i
- Una restricción/arista existe entre el nodo $\mathbf{x}_i$ y $\mathbf{x}_j$ si ...



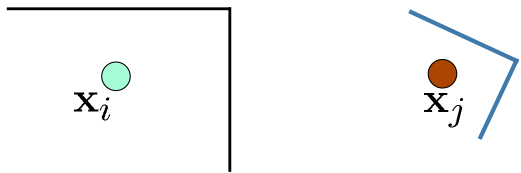
Crear una arista si...

- Una restricción/arista existe entre el nodo $\mathbf{x}_i$ y $\mathbf{x}_j$ si el robot se mueve de $\mathbf{x}_i$ a $\mathbf{x}_{i+1}$
- Las aristas corresponden a odometría



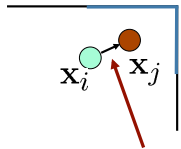
Crear una arista si...

- Una restricción/arista existe entre el nodo $\mathbf{x}_i$ y $\mathbf{x}_j$ si el robot observa la misma parte del entorno desde $\mathbf{x}_i$ y desde $\mathbf{x}_j$



Crear una arista si...

- Una restricción/arista existe entre el nodo $\mathbf{x}_i$ y $\mathbf{x}_j$ si el robot observa la misma parte del entorno desde $\mathbf{x}_i$ y desde $\mathbf{x}_j$
- Construimos una **restricción virtual** entre la posición de $\mathbf{x}_j$ vista desde $\mathbf{x}_i$

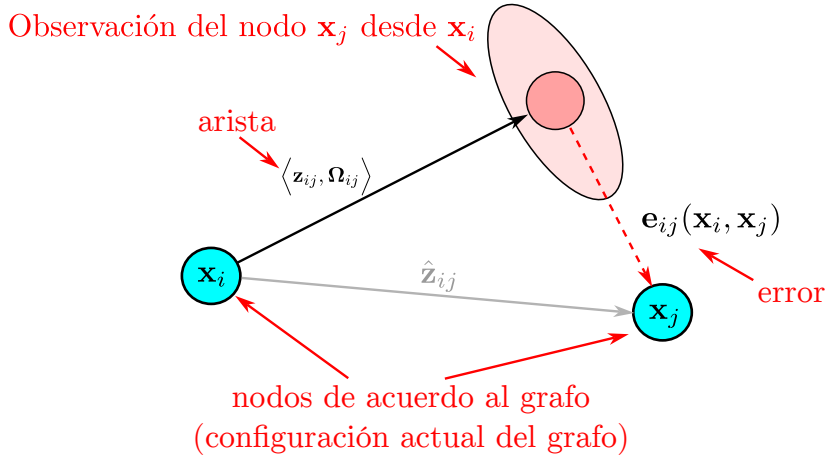


Edge represents the position of $\mathbf{x}_j$ seen from $\mathbf{x}_i$ based on the **observation**

La matriz de información de las aristas

- Las observaciones tienen ruido
- La matriz de información $\mathbf{\Omega}_{ij}$ para cada arista codifica su incertidumbre
- Mientras más grande $\mathbf{\Omega}_{ij}$, más importa la arista en al optimización.

La matriz de información de las aristas



Objetivo:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{ij} \mathbf{e}_{ij}^\top \mathbf{\Omega}_{ij} \mathbf{e}_{ij}$$

Mínimos cuadrados en SLAM

Podemos minimizar el error utilizando mínimos cuadrados (*Least Square*)

$$
\begin{aligned}
x^* &= \arg\min_{\mathbf{x}} \sum_{ij} \mathbf{e}_{ij}^\top(\mathbf{x}_i, \mathbf{x}_j)\mathbf{\Omega}_{ij}\mathbf{e}_{ij}(\mathbf{x}_i, \mathbf{x}_j) \\
&= \arg\min_{\mathbf{x}} \sum_{k} \mathbf{e}_{k}^\top(\mathbf{x})\mathbf{\Omega}_{k}\mathbf{e}_{k}(\mathbf{x})
\end{aligned}
$$

El **vector estado** es la concatenación de los nodos pose $\mathbf{x} = (\mathbf{x}_1^\top \mathbf{x}_2^\top \dots \mathbf{x}_n^\top)$. Cada nodo es una pose (posición u orientación).

Tenemos que definir la función de error...

Material para Least Squares

Terminar slides de Least Squares

- Cyrill Stachniss - Least Squares - An informal Introduction
  `https://youtu.be/r2cyMQ5NB1o?si=WYODHSkWun3FL7jR`

Least Squares en General

Enfoque para calcular una solución para un sistema sobredeterminado

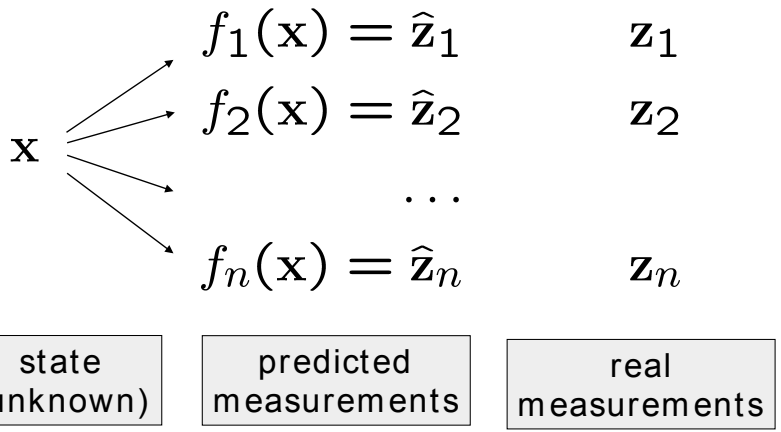
- Más ecuaciones que incógnitas
- Minimiza la suma de cuadrados de los errores en las ecuaciones
- Enfoque estándar para un gran conjunto de problemas
- Se utiliza para estimar el modelo de parámetros dado un conjunto de observaciones

Nuestro Problema

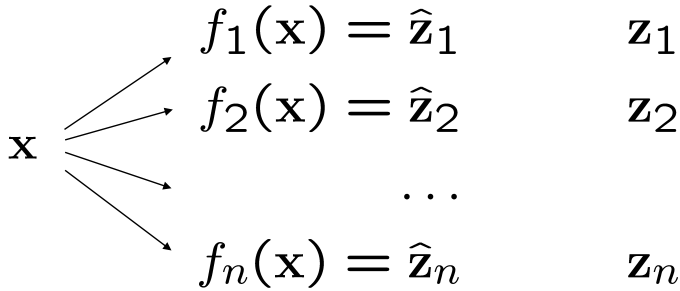
Dado un sistema descripto por un conjunto de n funciones de observación $\{f_i(x)\}_{i=1:n}$

- $\mathbf{x}$ es el vector de estado
- $\mathbf{z}$ es una una medición del estado $\mathbf{x}$
- $\hat{\mathbf{z}}_i = f_i(\mathbf{x})$ es una función que mapea el estado $\mathbf{x}$ a una medición predicha $\hat{\mathbf{z}}_i$
- Dadas n mediciones ruidosas $\mathbf{z}_{1:n}$ acerca del estado $\mathbf{x}$
- Objetivo: Estimar el estado $\mathbf{x}$ que mejor explica las mediciones $\mathbf{z}_{1:n}$

Explicación Gráfica



Ejemplo



- $\mathbf{x}$ posición de los puntos 3D
- $\mathbf{z}_i$ coordenadas de los puntos 3D proyectados en las imágenes
- Estimar la posición 3D más probable de los puntos basado en las proyecciones en las imágenes (dada las poses de la cámara)

Función de Error

- El error $\mathbf{e}_i$ suele ser la diferencia entre la medición real y la predicción:

$$\mathbf{e}_i\left(\mathbf{x}\right) = \mathbf{z}_i - f_i\left(\mathbf{x}\right)$$

- Supongamos que el error tiene una distribución normal con media cero
- Error Gaussiano con matriz de información $\mathbf{\Omega}_i$
- El error al cuadrado de una medición depende sólo del estado y es un escalar:

$$\mathbf{e}_i\left(\mathbf{x}\right) = \mathbf{e}_i\left(\mathbf{x}\right)^\top \mathbf{\Omega}_i \mathbf{e}_i\left(\mathbf{x}\right)$$

Objetivo: Encontrar el Mínimo

- Encontrar el estado $\mathbf{x}^*$ que minimiza el error dados todas las mediciones

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} F(\mathbf{x}) \quad \leftarrow \text{global error (scalar)}$$

$$= \underset{\mathbf{x}}{\operatorname{argmin}} \sum_i e_i(\mathbf{x}) \quad \leftarrow \text{squared error terms (scalar)}$$

$$= \underset{\mathbf{x}}{\operatorname{argmin}} \sum_i \mathbf{e}_i^T(\mathbf{x}) \mathbf{\Omega}_i \mathbf{e}_i(\mathbf{x})$$

$\uparrow$ error terms (vector)

Objetivo: Encontrar el Mínimo

- Encontrar el estado $\mathbf{x}^*$ que minimiza el error dados todas las mediciones

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_i \mathbf{e}_i \left(\mathbf{x}\right)^\top \mathbf{\Omega}_i \mathbf{e}_i \left(\mathbf{x}\right)$$

- Una solución general es derivar la función de error global y encontrar sus nulos
- En general compleja y con solución no cerrada $\rightarrow$ Solución con Métodos numéricos

Suposiciones

- Hay una “buena” solución inicial disponible
- Las funciones de error son “suaves” en la vecindad del mínimo (con suerte global)
- Entonces, podemos resolver el problema con linearizaciones locales iterativas

Resolvemos utilizando Linearizaciones locales iterativas

- Linealizar los términos de error alrededor del solución actual/solución inicial
- Calcular la primera derivada de la función de error al cuadrado
- Setear en cero y resolver el sistema lineal
- Obtener el nuevo estado (que con suerte estará más cerca del mínimo)
- Iterar

Linearizar la Función de Error

- Podemos aproximar el error al rededor de una estimación inicial $\mathbf{x}$ a través de una expansión de Taylor

$$\mathbf{e}_i(\mathbf{x}+\boldsymbol{\Delta}\mathbf{x}) \simeq \underbrace{\mathbf{e}_i(\mathbf{x})}_{\mathbf{e}_i} + \mathbf{J}_i\boldsymbol{\Delta}\mathbf{x} \quad \text{con} \quad \mathbf{J}_i = \frac{\partial \mathbf{e}_i(\mathbf{x})}{\partial \mathbf{x}}$$

Error Cuadrático

- Con la linealización anterior, podemos fijar $\mathbf{x}$ y llevar a cabo la minimización en los incrementos $\Delta\mathbf{x}$
- Reemplazamos la expansión de Taylor en los términos de error al cuadrado:

$$e_i(\mathbf{x} + \boldsymbol{\Delta}\mathbf{x}) = \dots$$

Error Cuadrático

- Con la linealización anterior, podemos fijar $\mathbf{x}$ y llevar a cabo la minimización en los incrementos $\Delta\mathbf{x}$
- Reemplazamos la expansión de Taylor en los términos de error al cuadrado:

$$e_i(\mathbf{x} + \boldsymbol{\Delta}\mathbf{x}) = \mathbf{e}_i \left(\mathbf{x} + \Delta\mathbf{x}\right)^\top \boldsymbol{\Omega}_i \mathbf{e}_i \left(\mathbf{x} + \Delta\mathbf{x}\right)$$

Error Cuadrático

- Con la linealización anterior, podemos fijar $\mathbf{x}$ y llevar a cabo la minimización en los incrementos $\Delta\mathbf{x}$
- Reemplazamos la expansión de Taylor en los términos de error al cuadrado:

$$\begin{aligned} e_i(\mathbf{x} + \boldsymbol{\Delta}\mathbf{x}) &= \mathbf{e}_i \left(\mathbf{x} + \Delta\mathbf{x}\right)^\top \boldsymbol{\Omega}_i \mathbf{e}_i \left(\mathbf{x} + \Delta\mathbf{x}\right) \\ &\simeq \left(\mathbf{e}_i + \mathbf{J}_i \Delta\mathbf{x}\right)^\top \boldsymbol{\Omega}_i \left(\mathbf{e}_i + \mathbf{J}_i \Delta\mathbf{x}\right) \end{aligned}$$

Error Cuadrático

- Con la linealización anterior, podemos fijar $\mathbf{x}$ y llevar a cabo la minimización en los incrementos $\Delta\mathbf{x}$
- Reemplazamos la expansión de Taylor en los términos de error al cuadrado:

$$\begin{aligned} e_i(\mathbf{x}+\boldsymbol{\Delta}\mathbf{x}) &= \mathbf{e}_i\left(\mathbf{x}+\Delta\mathbf{x}\right)^\top \boldsymbol{\Omega}_i \mathbf{e}_i\left(\mathbf{x}+\Delta\mathbf{x}\right) \\ &\simeq \left(\mathbf{e}_i+\mathbf{J}_i\Delta\mathbf{x}\right)^\top \boldsymbol{\Omega}_i\left(\mathbf{e}_i+\mathbf{J}_i\Delta\mathbf{x}\right) \\ &= \mathbf{e}_i^\top\boldsymbol{\Omega}_i\mathbf{e}_i + \mathbf{e}_i^\top\boldsymbol{\Omega}_i\mathbf{J}_i\Delta\mathbf{x} + \Delta\mathbf{x}^\top\mathbf{J}_i^\top\boldsymbol{\Omega}_i\mathbf{e}_i + \Delta\mathbf{x}^\top\mathbf{J}_i^\top\boldsymbol{\Omega}_i\mathbf{J}_i\Delta\mathbf{x} \end{aligned}$$

Error Cuadrático (cont.)

- Todos los sumandos son escalares por lo que la transposición no tiene ningún efecto
- Agrupando términos similares obtenemos:

$$e_i(\mathbf{x} + \boldsymbol{\Delta}\mathbf{x}) \simeq \mathbf{e}_i^\top \boldsymbol{\Omega}_i \mathbf{e}_i + \mathbf{e}_i^\top \boldsymbol{\Omega}_i \mathbf{J}_i \Delta\mathbf{x} + \Delta\mathbf{x}^\top \mathbf{J}_i^\top \boldsymbol{\Omega}_i \mathbf{e}_i + \Delta\mathbf{x}^\top \mathbf{J}_i^\top \boldsymbol{\Omega}_i \mathbf{J}_i \Delta\mathbf{x}$$

Error Cuadrático (cont.)

- Todos los sumandos son escalares por lo que la transposición no tiene ningún efecto
- Agrupando términos similares obtenemos:

$$
\begin{aligned}
e_i(\mathbf{x}+\boldsymbol{\Delta}\mathbf{x}) &\simeq \mathbf{e}_i^\top \boldsymbol{\Omega}_i \mathbf{e}_i + \mathbf{e}_i^\top \boldsymbol{\Omega}_i \mathbf{J}_i \Delta\mathbf{x} + \Delta\mathbf{x}^\top \mathbf{J}_i^\top \boldsymbol{\Omega}_i \mathbf{e}_i + \Delta\mathbf{x}^\top \mathbf{J}_i^\top \boldsymbol{\Omega}_i \mathbf{J}_i \Delta\mathbf{x} \\
&= \underbrace{\mathbf{e}_i^\top \boldsymbol{\Omega}_i \mathbf{e}_i}_{c_i} + 2 \underbrace{\mathbf{e}_i^\top \boldsymbol{\Omega}_i \mathbf{J}_i}_{\mathbf{b}_i^\top} \Delta\mathbf{x} + \Delta\mathbf{x}^\top \underbrace{\mathbf{J}_i^\top \boldsymbol{\Omega}_i \mathbf{J}_i}_{\mathbf{H}_i} \Delta\mathbf{x}
\end{aligned}
$$

Error Cuadrático (cont.)

- Todos los sumandos son escalares por lo que la transposición no tiene ningún efecto
- Agrupando términos similares obtenemos:

$$
\begin{aligned}
e_i(\mathbf{x}+\boldsymbol{\Delta}\mathbf{x}) &\simeq \mathbf{e}_i^\top \boldsymbol{\Omega}_i \mathbf{e}_i + \mathbf{e}_i^\top \boldsymbol{\Omega}_i \mathbf{J}_i \Delta\mathbf{x} + \Delta\mathbf{x}^\top \mathbf{J}_i^\top \boldsymbol{\Omega}_i \mathbf{e}_i + \Delta\mathbf{x}^\top \mathbf{J}_i^\top \boldsymbol{\Omega}_i \mathbf{J}_i \Delta\mathbf{x} \\
&= \underbrace{\mathbf{e}_i^\top \boldsymbol{\Omega}_i \mathbf{e}_i}_{c_i} + 2 \underbrace{\mathbf{e}_i^\top \boldsymbol{\Omega}_i \mathbf{J}_i}_{\mathbf{b}_i^\top} \Delta\mathbf{x} + \Delta\mathbf{x}^\top \underbrace{\mathbf{J}_i^\top \boldsymbol{\Omega}_i \mathbf{J}_i}_{\mathbf{H}_i} \Delta\mathbf{x} \\
&= c_i + 2\mathbf{b}_i^\top \Delta\mathbf{x} + \Delta\mathbf{x}^\top \mathbf{H}_i \Delta\mathbf{x}
\end{aligned}
$$

Error Global

- El error global es la suma de términos de errores cuadrados correspondientes a las medidas individuales
- Forma una nueva expresión, que se aproxima al error global en la vecindad de la solución actual $\mathbf{x}$

$$
\begin{aligned}
F(\mathbf{x}+\Delta\mathbf{x}) &= \sum_i e_i(\mathbf{x}+\boldsymbol{\Delta}\mathbf{x}) \\
F(\mathbf{x}+\Delta\mathbf{x}) &\simeq \sum_i \left(c_i + 2\mathbf{b}_i^\top \Delta\mathbf{x} + \Delta\mathbf{x}^\top \mathbf{H}_i \Delta\mathbf{x}\right) \\
&= \sum_i c_i + 2\left(\sum_i \mathbf{b}_i^\top\right)\Delta\mathbf{x} + \Delta\mathbf{x}^\top\left(\sum_i \mathbf{H}_i\right)\Delta\mathbf{x}
\end{aligned}
$$

Error Global (cont.)

$$
\begin{aligned}
F(\mathbf{x}+\Delta\mathbf{x}) &\simeq \sum_i \left(c_i + 2\mathbf{b}_i^\top \Delta\mathbf{x} + \Delta\mathbf{x}^\top \mathbf{H}_i \Delta\mathbf{x}\right) \\
&= \underbrace{\sum_i c_i}_{c} + 2\underbrace{\left(\sum_i \mathbf{b}_i^\top\right)}_{\mathbf{b}^\top}\Delta\mathbf{x} + \Delta\mathbf{x}^\top \underbrace{\left(\sum_i \mathbf{H}_i\right)}_{\mathbf{H}}\Delta\mathbf{x} \\
&= c + 2\mathbf{b}^\top\Delta\mathbf{x} + \Delta\mathbf{x}^\top\mathbf{H}\Delta\mathbf{x}
\end{aligned}
$$

con

$$
\begin{aligned}
\mathbf{b}^\top &= \sum_i \mathbf{e}_i^\top \mathbf{\Omega}_i \mathbf{J}_i \\
\mathbf{H} &= \sum_i \mathbf{J}_i^\top \mathbf{\Omega}_i \mathbf{J}_i
\end{aligned}
$$

Forma Cuadrática

- Podemos escribir los términos de error global como una forma cuadrática en $\Delta\mathbf{x}$

$$F(\mathbf{x} + \Delta\mathbf{x}) = c + 2\mathbf{b}^\top \Delta\mathbf{x} + \Delta\mathbf{x}^\top \mathbf{H} \Delta\mathbf{x}$$

- ¿Cómo calcular el mínimo de una forma cuadrática?

Forma Cuadrática

- Podemos escribir los términos de error global como una forma cuadrática en $\Delta\mathbf{x}$

$$F(\mathbf{x} + \Delta\mathbf{x}) = c + 2\mathbf{b}^\top \Delta\mathbf{x} + \Delta\mathbf{x}^\top \mathbf{H} \Delta\mathbf{x}$$

- Calcular la derivada de $F(\mathbf{x} + \Delta\mathbf{x})$ con respecto a $\Delta\mathbf{x}$ (dado $\mathbf{x}$)
- Derivamos e igualamos a 0
- resolvemos

Derivando la forma cuadrática

- Dada la forma cuadrática

$$f(\mathbf{x}) = \mathbf{x}^\top \mathbf{H}\mathbf{x} + \mathbf{b}^\top \mathbf{x}$$

- La primera derivada es

$$\frac{\partial f}{\partial \mathbf{x}} = \left(\mathbf{H} + \mathbf{H}^\top\right)\mathbf{x} + \mathbf{b}$$

Ver: The Matrix Cookbook, sección 2.4.2

Forma Cuadrática

- Podemos escribir los términos de error global de forma cuadrática en $\Delta\mathbf{x}$

$$F(\mathbf{x}+\Delta\mathbf{x}) = c + 2\mathbf{b}^{\top}\Delta\mathbf{x} + \Delta\mathbf{x}^{\top}\mathbf{H}\Delta\mathbf{x}$$

- La derivada de $F(\mathbf{x}+\Delta\mathbf{x})$

$$\frac{\partial F(\mathbf{x}+\Delta\mathbf{x})}{\partial\Delta\mathbf{x}} \simeq 2\mathbf{b} + 2\mathbf{H}\Delta\mathbf{x}$$

Minimizando la forma cuadrática

- Derivada de $F(\mathbf{x} + \Delta\mathbf{x})$

$$\frac{\partial F(\mathbf{x} + \Delta\mathbf{x})}{\partial \Delta\mathbf{x}} \simeq 2\mathbf{b} + 2\mathbf{H}\Delta\mathbf{x}$$

- Igualando a 0

$$0 = 2\mathbf{b} + 2\mathbf{H}\Delta\mathbf{x}$$

- Lo que lleva al sistema lineal

$$\mathbf{H}\Delta\mathbf{x} = -\mathbf{b}$$

- La solución para el incremento $\Delta\mathbf{x}^*$ es

$$\Delta\mathbf{x}^* = -\mathbf{H}^{-1}\mathbf{b}$$

Solución con Gauss-Newton

Iterar los siguientes pasos:

- Lineanizar cerca de $\mathbf{x}$ y computar para cada medición

$$\mathbf{e}_i(\mathbf{x} + \boldsymbol{\Delta}\mathbf{x}) \simeq \mathbf{e}_i(\mathbf{x}) + \mathbf{J}_i \boldsymbol{\Delta}\mathbf{x}$$

- Computar los términos del sistema lineal

$$\mathbf{b}^\top = \sum_i \mathbf{e}_i^\top \boldsymbol{\Omega}_i \mathbf{J}_i \qquad \mathbf{H} = \sum_i \mathbf{J}_i^\top \boldsymbol{\Omega}_i \mathbf{J}_i$$

- Resolver el sistema lineal

$$\Delta\mathbf{x}^* = -\mathbf{H}^{-1}\mathbf{b}$$

- Actualizar el estado

$$\mathbf{x} \leftarrow \mathbf{x} + \Delta\mathbf{x}^*$$

Ejemplo: Calibración de Odometría

- Mediciones de Odometría u_i
- Eliminar el error sistemático a través de la calibración
- Suposición: disponemos del ground-truth de odometría u_i^*
- Ground-truth dado por sistemas: Motion Capture (Vicon), Scan-Matching o SLAM

Ejemplo: Calibración de Odometría

- Hay una función que $f_i(\mathbf{x})$ que dado algunos parámetros de bias $\mathbf{x}$, devuelve una odometría corregida (*unbiased*) para la lectura ruidosa u_i'

$$u_i' = f_i(\mathbf{x}) = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix} u_i$$

- Para obtener la función de corrección $f(\mathbf{x})$, necesitamos encontrar los parámetros $\mathbf{x}$

Calibración de Odoemtría (cont.)

- El vector estado es

$$\mathbf{x} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{21} & x_{22} & x_{23} & x_{31} & x_{32} & x_{33} \end{bmatrix}$$

- La función error es

$$\mathbf{e}_i(\mathbf{x}) = u_i^* - \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix} u_i$$

- Su derivada es

$$\mathbf{J}_i = \frac{\partial \mathbf{e}_i(\mathbf{x})}{\partial \mathbf{x}} = - \begin{pmatrix} u_{i,x} & u_{i,y} & u_{i,\theta} & & & & & & \\ & & & u_{i,x} & u_{i,y} & u_{i,\theta} & & & \\ & & & & & & u_{i,x} & u_{i,y} & u_{i,\theta} \end{pmatrix}$$

Does not depend on ***x***, why? What are the consequences? ➡ ***e*** is linear, no need to iterate!

Que la primera derivada (Jacobiano) no dependa de $\mathbf{x}$, es lo menos común, ya que significa que la función $\mathbf{e}$ es lineal!
En este caso solo vamos a tener que hacer una sola iteración.

Preguntas

- ¿Cómo lucen los parámetros si la odometría es perfecta?
- ¿Cuantas mediciones son necesarias para encontrar la solución al problema de calibración?
- $\mathbf{H}$ es simétrica. ¿Por qué?
- ¿Cómo afecta la estructura de la función de medición a la estructura de $\mathbf{H}$?

¿Cómo Resolver de manera eficiente un Sistema Lineal?

- Sistema Lineal $\mathbf{H}\Delta\mathbf{x} = -\mathbf{b}$
- Podemos resolverlo utilizando inversión de matrices (en teoría)
- En la práctica:
  - Factorización de Cholesky
  - Descomposición QR
  - Métodos iterativos como el método del Gradiente Conjugado (para sistemas grandes)

Descomposición de Cholesky para Resolver un Sistema Lineal

- Sea la matriz $\mathbf{A}$ simétrica positiva definida
- El sistema a resolver es $\mathbf{Ax} = \mathbf{b}$
- Cholesky lleva a $\mathbf{A} = \mathbf{LL}^\top$ con $\mathbf{L}$ matriz triangular inferior

Descomposición de Cholesky para Resolver un Sistema Lineal

- Sea la matriz $\mathbf{A}$ simétrica positiva definida
- El sistema a resolver es $\mathbf{Ax} = \mathbf{b}$
- Cholesky lleva a $\mathbf{A} = \mathbf{LL}^\top$ con $\mathbf{L}$ matriz triangular inferior
- Resolvemos primero

$$\mathbf{Ly} = \mathbf{b}$$

- y luego,

$$\mathbf{L}^\top \mathbf{x} = \mathbf{y}$$

Resumen de Gauss-Newton

Método para minimizar un error al cuadrado:

- Comenzar con una solución inicial (*initial guess*)
- Linealizar las funciones de error individuales
- Esto lleva a una forma cuadrática
- Se obtiene un sistema lineal derivando e igualando a 0
- Resolver el sistema lineale conduce a una actualización del estado
- Iterar

Least Squares vs. Probabilistic State Estimation

- Hasta ahora, minimizamos una función de error
- ¿Cómo se relaciones esto con la estimación de estado en el sentido probabilístico?

Comencemos con Estimación de estado

- Regla de Bayes, suposiciones de independencia y Markov nos permiten reescribir

$$p\left(x_{0:t}|z_{1:t},u_{1:t}\right)=\eta p\left(x_0\right)\prod_t\left[p\left(x_t|x_{t-1},u_t\right)p\left(z_t|x_t\right)\right]$$

Log Likelihood

- Reescribiendo como el log likelihood, lleva a

$$\log p\left(x_{0:t}|z_{1:t}, u_{1:t}\right) = \text{const.} \log p\left(x_0\right) + \sum_t \left[\log p\left(x_t|x_{t-1}, u_t\right) + \log p\left(z_t|x_t\right)\right]$$

Suposición Gaussiana

- Suponiendo distribución Gaussiana

$$\log p\left(x_{0:t}|z_{1:t}, u_{1:t}\right) = \text{const.} \log \underbrace{p\left(x_0\right)}_{\mathcal{N}} + \sum_t \left[\log \underbrace{p\left(x_t|x_{t-1}, u_t\right)}_{\mathcal{N}} + \log \underbrace{p\left(z_t|x_t\right)}_{\mathcal{N}} \right]$$

Log de una Gaussiana

- La función de distribución de la distribución normal está definida como

$$p(x) = \det(2\pi\Sigma)^{\frac{1}{2}} \exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right)$$

- Log likelihood de una Gaussiana

$$\log \mathcal{N}(x, \mu, \Sigma) = \text{const.} - \frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)$$

Función de Error como Exponente

- Log likelihood de una Gaussiana

$$\log \mathcal{N}(x, \mu, \Sigma) = \text{const.} - \frac{1}{2} \underbrace{\underbrace{(x-\mu)^\top}_{\mathbf{e}^\top(x)} \underbrace{\Sigma^{-1}}_{\mathbf{\Omega}} \underbrace{(x-\mu)}_{\mathbf{e}(x)}}_{e(x)}$$

- está a una constante de equivalencia de las funciones de error usadas antes

Log Likelihood con Términos de Error

- Suponiendo distribución Gaussiana

$$\log p\left(x_{0:t}|z_{1:t}, u_{1:t}\right) = \text{const.} - \frac{1}{2}e_p(x) - \frac{1}{2}\sum_t \left[e_{u_t}(x) + e_{z_t}(x)\right]$$

Maximizing el Log Likelihood

- Suponiendo distribución Gaussiana

$$\log p\left(x_{0:t}|z_{1:t}, u_{1:t}\right) = \text{const.} - \frac{1}{2}e_p(x) - \frac{1}{2}\sum_t \left[e_{u_t}(x) + e_{z_t}(x)\right]$$

- Maximizando el log likelihood lleva a

$$\arg\max \log p\left(x_{0:t}|z_{1:t}, u_{1:t}\right) = \arg\min e_p(x) + \sum_t \left[e_{u_t}(x) + e_{z_t}(x)\right]$$

Minimizar el error cuadrático es equivalente a Maximizar el Log Likelihood de Distribuciones Gaussianas Independientes

- Con términos de error individuales para los controles, mediciones y un prior:

$$\arg\max \log p\left(x_{0:t}|z_{1:t}, u_{1:t}\right) = \arg\min e_p(x) + \sum_t \left[e_{u_t}(x) + e_{z_t}(x)\right]$$

Resumen

- Técnica para minimizar funciones de error cuadráticas
- Gauss-Newton es un enfoque iterativo para problemas no lineales
- Utiliza linearización (¡aproximación!)
- Equivalente a maximizar el log likelihood de Gaussianas independientes
- Método popular en muchas disciplinas.

Bibliografía de Gauss-Newton

- Capítulo 11.4 de [5]

Función de error

- Función de error para una sola restricción

$$\mathbf{e}_{ij}(x_i, x_j) = t2v(\mathbf{z}_{ij}^{-1}(\mathbf{x}_i^{-1}\mathbf{x}_j))$$

measurement

$\mathbf{x}_j$ referenciada con respecto a $\mathbf{x}_i$

- Error como una función de un vector estado completo

$$\mathbf{e}_{ij}(x) = t2v(\mathbf{z}_{ij}^{-1}(\mathbf{x}_i^{-1}\mathbf{x}_j))$$

- El error toma el valor 0 cuando

$$\mathbf{z}_{ij} = (\mathbf{x}_i^{-1}\mathbf{x}_j)$$

[1] $t2v(.)$ mapea transformaciones a vectores

Linearizar la función de error

- Podemos aproximar el error al rededor de una estimación inicial $\mathbf{x}$ a través de una expansión de Taylor

$$\mathbf{e}_{ij}(\mathbf{x}+\mathbf{\Delta x}) \simeq \mathbf{e}_{ij}(\mathbf{x}) + \mathbf{J}_{ij}\mathbf{\Delta x} \quad \text{con} \quad \mathbf{J}_{ij} = \frac{\partial \mathbf{e}_{ij}(\mathbf{x})}{\partial \mathbf{x}}$$

Derivada de la función de error

- Pregunta: ¿Depende un término de error $\mathbf{e}_{ij}(\mathbf{x})$ de todas las variables de estado?

Derivada de la función de error

- Pregunta: ¿Depende un término de error $\mathbf{e}_{ij}(\mathbf{x})$ de todas las variables de estado?
  No, solo depende de $\mathbf{x}_i$ y $\mathbf{x}_j$

Derivada de la función de error

- Pregunta: ¿Depende un término de error $\mathbf{e}_{ij}(\mathbf{x})$ de todas las variables de estado? No, solo depende de $\mathbf{x}_i$ y $\mathbf{x}_j$
- Pregunta: ¿Hay alguna consecuencia en la estructura del Jacobiano?

Derivada de la función de error

- Pregunta: ¿Depende un término de error $\mathbf{e}_{ij}(\mathbf{x})$ de todas las variables de estado?
  No, solo depende de $\mathbf{x}_i$ y $\mathbf{x}_j$
- Pregunta: ¿Hay alguna consecuencia en la estructura del Jacobiano?
  Si, será diferente a cero solamente en las filas correspondientes a $\mathbf{x}_i$ y $\mathbf{x}_j$

$$\frac{\partial \mathbf{e}_{ij}(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} 0 & \dots & \dfrac{\partial \mathbf{e}_{ij}(\mathbf{x}_i)}{\partial \mathbf{x}_i} & \dots & 0 & \dots & \dfrac{\partial \mathbf{e}_{ij}(\mathbf{x}_j)}{\partial \mathbf{x}_j} & \dots & 0 \end{bmatrix}$$

$$\mathbf{J}_{ij} = \begin{bmatrix} 0 & \dots & \mathbf{A}_{ij} & \dots & 0 & \dots & \mathbf{B}_{ij} & \dots & 0 \end{bmatrix}$$

Jacobianos y problema disperso

- El error $\mathbf{e}_{ij}(\mathbf{x})$ depende solamente de los bloques de parámetros $\mathbf{x}_i$ y $\mathbf{x}_j$

$$\mathbf{e}_{ij}(\mathbf{x}) = \mathbf{e}_{ij}(\mathbf{x}_i, \mathbf{x}_j)$$

- El Jacobiano será cero en todos lados excepto en las columnas $\mathbf{x}_i$ y $\mathbf{x}_j$

$$\mathbf{J}_{ij} = \begin{bmatrix} 0 & \dots & 0 & \dfrac{\partial \mathbf{e}_{ij}(\mathbf{x}_i)}{\partial \mathbf{x}_i} & 0 & \dots & 0 & \dfrac{\partial \mathbf{e}_{ij}(\mathbf{x}_j)}{\partial \mathbf{x}_j} & 0 & \dots & 0 \end{bmatrix}$$

Esto nos permite resolver SLAM de manera eficiente!

Consecuencia de que sea un problema disperso

- Necesitamos computar el vector coeficiente

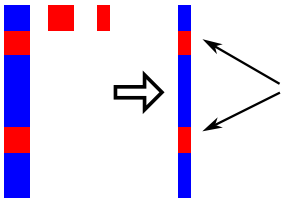
$$\mathbf{b}^\top = \sum_{ij} \mathbf{b}_{ij}^\top = \sum_{ij} \mathbf{e}_{ij}^\top \Omega_{ij} \mathbf{J}_{ij}$$

$$\mathbf{H} = \sum_{ij} \mathbf{H}_{ij} = \sum_{ij} \mathbf{J}_{ij}^\top \Omega_{ij} \mathbf{J}_{ij}$$

- La estructura dispersa de $\mathbf{J}_{ij}$ resultará en una estructura dispersa de $\mathbf{H}$
- Esta estructura refleja la matriz de adyacencia del grafo

Ilustración de la estructura

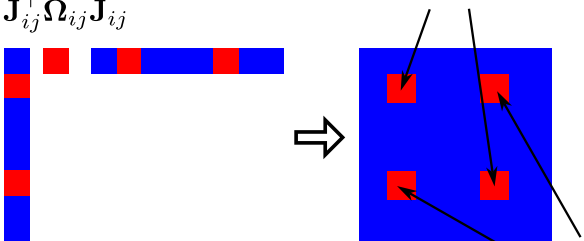
$$\mathbf{b}_{ij} = \mathbf{J}_{ij}^{\top} \mathbf{\Omega}_{ij} \mathbf{e}_{ij}$$



No ceros solamente en $\mathbf{x}_i$ y $\mathbf{x}_j$

$$\mathbf{H}_{ij} = \mathbf{J}_{ij}^{\top} \mathbf{\Omega}_{ij} \mathbf{J}_{ij}$$

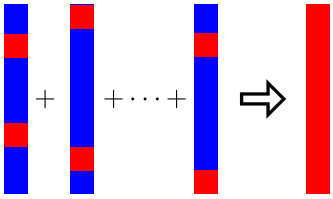
No ceros en la diagonal principal en $\mathbf{x}_i$ y $\mathbf{x}_j$



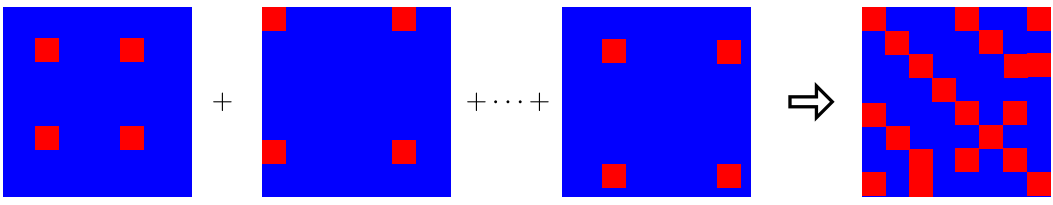
No ceros en los bloques ij y ji

Consecuencia de que sea un problema disperso

$$\mathbf{b} = \sum_{ij} \mathbf{b}_{ij}$$



$$\mathbf{H} = \sum_{ij} \mathbf{H}_{ij}$$



Consecuencia de que sea un problema disperso

- Una arista contribuye al sistema lineal a través de $\mathbf{b}_{ij}$ y $\mathbf{H}_{ij}$
- El vector coeficiente es

$$\begin{aligned}\mathbf{b}_{ij}^{\top} &= \mathbf{e}_{ij}^{\top}\mathbf{\Omega}_{ij}\mathbf{J}_{ij} \\ &= \mathbf{e}_{ij}^{\top}\mathbf{\Omega}_{ij}\begin{bmatrix}0 & \dots & \mathbf{A}_{ij} & \dots & \mathbf{B}_{ij} & \dots & 0\end{bmatrix} \\ &= \begin{bmatrix}0 & \dots & \mathbf{e}_{ij}^{\top}\mathbf{\Omega}_{ij}\mathbf{A}_{ij} & \dots & \mathbf{e}_{ij}^{\top}\mathbf{\Omega}_{ij}\mathbf{B}_{ij} & \dots & 0\end{bmatrix}\end{aligned}$$

- Es diferente a cero solamente en los índices correspondientes de $\mathbf{x}_i$ y $\mathbf{x}_j$

Consecuencia de que sea un problema disperso

- La matriz coeficiente de una arista es

$$\begin{aligned}\mathbf{H}_{ij}^{\top} &= \mathbf{J}_{ij}^{\top}\boldsymbol{\Omega}_{ij}\mathbf{J}_{ij} \\ &= \begin{bmatrix} \vdots \\ \mathbf{A}_{ij}^{\top} \\ \vdots \\ \mathbf{B}_{ij}^{\top} \\ \vdots \end{bmatrix} \boldsymbol{\Omega}_{ij} \begin{bmatrix} \cdots & \mathbf{A}_{ij} & \cdots & \mathbf{B}_{ij} & \cdots \end{bmatrix} \\ &= \begin{bmatrix} 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \mathbf{A}_{ij}^{\top}\boldsymbol{\Omega}_{ij}\mathbf{A}_{ij} & \vdots & \mathbf{A}_{ij}^{\top}\boldsymbol{\Omega}_{ij}\mathbf{B}_{ij} & \vdots \\ 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \mathbf{B}_{ij}^{\top}\boldsymbol{\Omega}_{ij}\mathbf{A}_{ij} & \vdots & \mathbf{B}_{ij}^{\top}\boldsymbol{\Omega}_{ij}\mathbf{B}_{ij} & \vdots \\ 0 & \cdots & 0 & \cdots & 0 \end{bmatrix}\end{aligned}$$

- Es diferente a cero en los bloques relacionados con i, j

Resumen del problema disperso

- Una arista ij solo contribuye a
  - i-ésimo y j-ésimo bloques de $\mathbf{b}_{ij}$
  - los bloques ii, jj, ij y ji de $\mathbf{H}_{ij}$
- El sistema resultante es disperso
- El sistema se puede computar sumando las contribuciones de cada arista
- Se pueden utilizar diferentes *solvers*
  - Descomposición de Sparse Cholesky
  - Gradiente conjudado
  - muchos más...

El sistema Lineal

- Vector de incrementos de estado

$$\Delta \mathbf{x}^\top = \begin{bmatrix} \Delta \mathbf{x}_1^\top & \Delta \mathbf{x}_2^\top & \cdots & \Delta \mathbf{x}_n^\top \end{bmatrix}$$

- Vector coeficiente

$$\mathbf{b}^\top = \begin{bmatrix} \overline{\mathbf{b}}_1^\top & \overline{\mathbf{b}}_2^\top & \cdots & \overline{\mathbf{b}}_n^\top \end{bmatrix}$$

- Matriz de ecuaciones normales

$$\mathbf{H} = \begin{bmatrix} \overline{\overline{\mathbf{H}}}_{11} & \overline{\overline{\mathbf{H}}}_{12} & \cdots & \overline{\overline{\mathbf{H}}}_{1n} \\ \overline{\overline{\mathbf{H}}}_{21} & \overline{\overline{\mathbf{H}}}_{22} & \cdots & \overline{\overline{\mathbf{H}}}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \overline{\overline{\mathbf{H}}}_{n1} & \overline{\overline{\mathbf{H}}}_{n2} & \cdots & \overline{\overline{\mathbf{H}}}_{nn} \end{bmatrix}$$

Construcción del sistema lineal

Para cada restricción:

- Computar el error $\mathbf{e}_{ij} = t2v(\mathbf{z}_{ij}^{-1}(\mathbf{x}_i^{-1}\mathbf{x}_j))$
- Computar los bloques de los jacobianos:

$$\mathbf{A}_{ij} = \frac{\partial \mathbf{e}_{ij}(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_i} \qquad \mathbf{B}_{ij} = \frac{\partial \mathbf{e}_{ij}(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_j}$$

- Actualizar el vector coeficiente:

$$\overline{\mathbf{b}}_i^\top += \mathbf{e}_{ij}^\top \mathbf{\Omega}_{ij} \mathbf{A}_{ij} \qquad \overline{\mathbf{b}}_j^\top += \mathbf{e}_{ij}^\top \mathbf{\Omega}_{ij} \mathbf{B}_{ij}$$

- Actualizar el vector coeficiente:

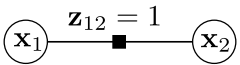
$$\overline{\mathbf{H}}_{ij}^\top += \mathbf{A}_{ij}^\top \mathbf{\Omega}_{ij} \mathbf{A}_{ij} \qquad \overline{\mathbf{H}}_{ij}^\top += \mathbf{A}_{ij}^\top \mathbf{\Omega}_{ij} \mathbf{B}_{ij}$$
$$\overline{\mathbf{H}}_{ij}^\top += \mathbf{B}_{ij}^\top \mathbf{\Omega}_{ij} \mathbf{A}_{ij} \qquad \overline{\mathbf{H}}_{ij}^\top += \mathbf{B}_{ij}^\top \mathbf{\Omega}_{ij} \mathbf{B}_{ij}$$

Algoritmo

1: **procedure** optimize($\mathbf{x}$)
2: **while** (**do**!converged)
3: $(\mathbf{H}, \mathbf{b}) = buildLinearSystem(\mathbf{x})$
4: $\mathbf{\Delta x} = solveSparse(\mathbf{H\Delta x} = -\mathbf{b})$
5: $\mathbf{x} = \mathbf{x} + \mathbf{\Delta x}$
6: **end while**
7: **return** $\mathbf{x}$
8: **end procedure**

Ejemplo Trivial 1D

Dos nodos y una observación



$$
\begin{aligned}
\mathbf{x} &= \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 \end{bmatrix}^\top = \begin{bmatrix} 0 & 0 \end{bmatrix} \\
\mathbf{z}_{12} &= 1 \\
\mathbf{\Omega}_{12} &= 2 \\
\mathbf{e}_{12} &= \mathbf{z}_{12} - (\mathbf{x}_2 - \mathbf{x}_1) = 1 - (0 - 0) = 1 \\
\mathbf{J}_{12} &= \begin{bmatrix} 1 & -1 \end{bmatrix} \\
\mathbf{b}_{12}^\top &= \mathbf{e}_{12}^\top \mathbf{\Omega}_{12} \mathbf{J}_{12} = \begin{bmatrix} 2 & -2 \end{bmatrix} \\
\mathbf{H}_{12} &= \mathbf{J}_{12}^\top \mathbf{\Omega}_{12} \mathbf{J}_{12} = \begin{bmatrix} 2 & -2 \\ -2 & 2 \end{bmatrix} \\
\mathbf{\Delta x} &= -\mathbf{H}_{12}^{-1} \mathbf{b}_{12}
\end{aligned}
$$

Problema: $\det(\mathbf{H}) = 0$, por lo tanto no podemos invertir $\mathbf{H}$

¿Qué es lo que anda mal?

- La restricción especifica una restricción relativa entre ambos nodos
- Cualquier pose de los nodos va a estar bien si se cumple la restricción relativa entre ellos. Este problema se lo conoce como **Gauge Freedom**
- Para resolverlo tenemos que **fijar** un nodo. Al fijar un nodo estamos agregando un **Prior**!

$$\mathbf{H} = \begin{bmatrix} 2 & -2 \\ -2 & 2 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \leftarrow \text{restricción que pone } \boldsymbol{\Delta}\mathbf{x}_1 = 0$$

$$\boldsymbol{\Delta}\mathbf{x} = -\mathbf{H}_{12}^{-1}\mathbf{b}_{12}$$

$$\boldsymbol{\Delta}\mathbf{x} = \begin{bmatrix} 0 & 1 \end{bmatrix}^\top$$

Con esto hacemos que el update del nodo $\mathbf{x}_1$ sea 0 y que el update de $\mathbf{x}_2$ sea actualizado con 1.

Rol del Prior

- Vimos que la matriz de información $\mathbf{H}$ no es de rango completa, y por tanto no es invertible
- Un marco de referencia global no ha sido fijado
- Fijar un marco de referencia global está fuertemente relacionado con el prior $p(\mathbf{x}_0)$
- Una estimación Gaussiana de $\mathbf{x}_0$ resulta en agregar una restricción
- Ejemplo: La primera pose debe estar fija en el origen de coordenadas

$$\mathbf{e}(\mathbf{x}_0) = t2v(\mathbf{x}_0)$$

Esto es, al minimizar, el término de error $\mathbf{e}(\mathbf{x}_0)$ queremos que sea lo más chico posible. Esto sucede cuando $\mathbf{x}_0$ tiende a 0.

Fijando un subconjunto de variables

- Asumimos que el valor de ciertas variables durante la optimización es conocido a priori
- Queremos optimizar todos las demás variables pero manteniendo estas como fijas
- Podemos hacer como hicimos con el prior anterior, pero el problema es que es solo una **soft constraint** y no es realmente un fix. Ya que puede suceder que otra constraint mueva a este nodo, y por tanto no podemos asegurar que tenga un valor fijo.
- **Para efectivamente hacer que una variable quede fija, tenemos que hacer que no sea optimizada y por tanto debemos removerla del sistema lineal**. Removerla del sistema lineal hace que no sea actualizada y que todas las demás variables esten restringidas por esta. Esto se hace construyendo todo el sistema lineal y luego sencillamente suprimiendo la fila y columna correspondiente a la variable. Finalmente se resuelve el sistema lineal.
- La supresión de la columna y fila funciona como una **condicionamiento**, es decir, es una condición que dice "Dado que un nodo toma un valor, las demás se ven afectadas de esta manera"

Podemos suprimir las columnas y las filas de las variables correspondientes

- El porqué podemos hacer esto, resulta del **condicionamiento de distribuciones Gaussianas**.
- **Condicionamiento en el espacio de información**, significa que podemos remover una parte de la matriz de información y esto corresponde una operación de condicionamiento de la distribución Gaussiana.
- La $\mathbf{H}$ es una matriz de información de todo nuestro problema con todas las restricciones juntas. Por tanto, removiendo una fila y una columna estamos condicionando al sistema haciendo que estas esten fijas y no se puedan actualizar y todas las demás variables queden restringidas por estas.

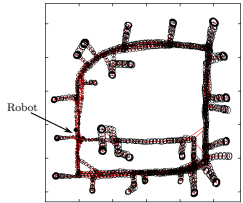
[1] Mas info en paper: Exactly sparse delayed-state filters for view-based SLAM. Eustice, Ryan M., Singh, Hanumant, Leonard, John J.

Incertidumbre

- $\mathbf{H}$ representa la matriz de información dado el punto de linearización
- La inversa de $\mathbf{H}$ es la matriz de covarianza (densa). Calcular la inversa es muy costoso computacionalmente. Sin embargo, podemos computar partes de la matriz de covarianza.
- Los bloques de la diagonal de la matriz de covarianza representan las incertidumbres de las variables correspondientes

Incertidumbre relativa

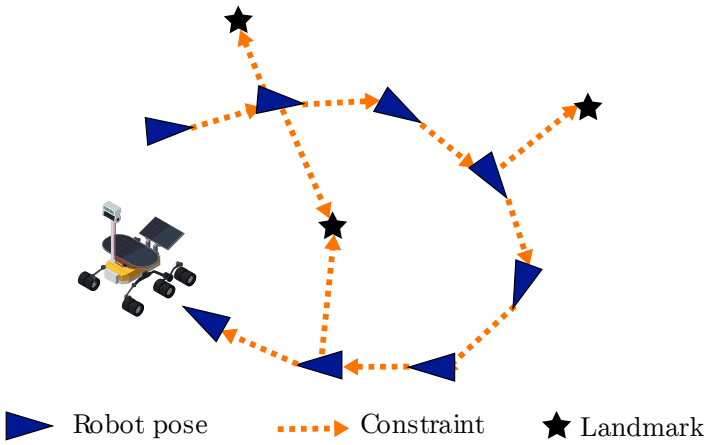
- Determinar la incertidumbre relativa entre $\mathbf{x}_i$ y $\mathbf{x}_j$.
  - Esto es especialmente útil para la detección de ciclos, ya que podemos computar la incertidumbre de la pose actual con respecto a una pose anterior. De manera de poder determinar si es posible que haya un ciclo entre las poses o no (si la pose anterior se encuentra dentro de la covarianza entonces detectamos un ciclo).
- Para determina la incertidumbre relativa entre $\mathbf{x}_i$ y $\mathbf{x}_j$ vamos a:
  - Construir la matriz completa $\mathbf{H}$
  - Suprimir las filas y columnas de $\mathbf{x}_i$ (equivalente a no optimizar/fijar la variable)
  - Computar el bloque j, j de la inversa
  - Este bloque contiene la matriz de covarianza de $\mathbf{x}_j$ con respecto a $\mathbf{x}_i$ la cuál está fija.



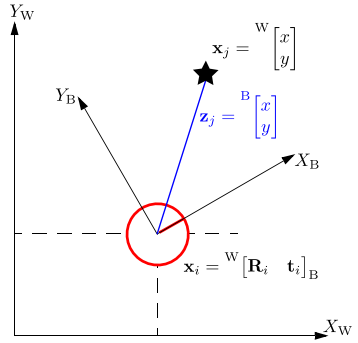
Resumen de Pose-Graph

- El back-end del problema de SLAM puede ser resuelto aplicando Gauss-Newton
- La matriz $\mathbf{H}$ es dispersa
- Que $\mathbf{H}$ sea dispersa nos permite resolver el sistema lineal de manera eficiente

Graph-Based SLAM con landmarks



Observaciones de landmarks



- Observación esperada para un (x-y sensor[2])

$$\hat{\mathbf{z}}_{ij}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{R}_i^\top (\mathbf{x}_j - \mathbf{t}_i)$$

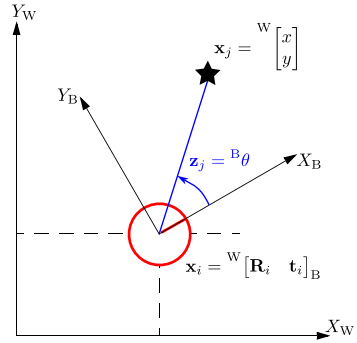
robot landmark robot translation

- Función error

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

[2] x-y sensor es un sensor cuyas mediciones tiene la forma (x,y), un punto de la escena. En un mundo 2D.

Bearing only obsevations (observaciones de ángulo)



- Un landmark es un punto 2D
- El robot observa el ángulo hacia el landmark
- Función de observación

robot-landmark ángulo

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

robot landmark robot orientation

- Función error

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

Rango de la matriz H

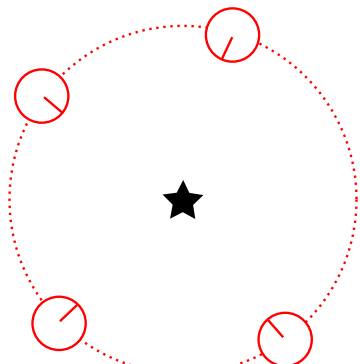
- ¿Cuál es el rango de la matriz $\mathbf{H}_{ij}$ para una restricción pose-landmark 2D?
  - El rango de $\mathbf{H}_{ij}$ está determinado por el rango del Jacobiano $\mathbf{J}_{ij}$ que a lo sumo es una matriz de 2×5 (2 porque la medición da información 2D y 5 por $\begin{bmatrix} x & y & \theta & l_x & l_y \end{bmatrix}$)
  - $\mathbf{H}_{ij}$ no puede tener más que rango 2
    $rank(\mathbf{H}_{ij}) = rank(\mathbf{J}_{ij}^{\top} \mathbf{\Omega}_{ij} \mathbf{J}_{ij}) = rank(\mathbf{J}_{ij})$ Ver: The Matrix Cookcook sec. 9.6.9
- ¿Cuál es el rango de la matriz $\mathbf{H}_{ij}$ para una restricción pose-landmark bearing-only?
  - El rango de $\mathbf{H}_{ij}$ está determinado por el rango del Jacobiano $\mathbf{J}_{ij}$ que a lo sumo es una matriz de 1×5. 1 porque la medición da información 1D y 5 por $\begin{bmatrix} x & y & \theta & l_x & l_y \end{bmatrix}$
  - $\mathbf{H}_{ij}$ tiene rango 1

¿Dónde está el robot?

- El robot observa un landmark (x, y)
- ¿Dónde puede estar el robot relativo al landmark?

¿Dónde está el robot?

- El robot observa un landmark (x, y)
- ¿Dónde puede estar el robot relativo al landmark?



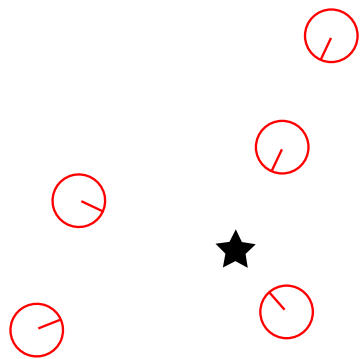
El robot puede estar en cualquier lugar del círculo.
Es un espacio de soluciones 1D (restringido por la distancia y la orientación del robot)

¿Dónde está el robot?

- El robot observa un landmark (bearing-only)
- ¿Dónde puede estar el robot relativo al landmark?

¿Dónde está el robot?

- El robot observa un landmark (bearing-only)
- ¿Dónde puede estar el robot relativo al landmark?



El robot puede estar en cualquier lugar del plano xy. Siempre estará apuntando hacia el landmark. Es un espacio de soluciones 2D (restringido por la orientación del robot).

Rango

- En SLAM con landmarks, el sistema puede quedar indeterminado
- El rango de $\mathbf{H}$ es **menor o igual** que la suma de los rangos de todas las observaciones
- Para determinar una **solución única**, el sistema debe tener **rango completo**

Rango

- En SLAM con landmarks, el sistema puede quedar indeterminado
- El rango de $\mathbf{H}$ es **menor o igual** que la suma de los rangos de todas las observaciones
- Para determinar una **solución única**, el sistema debe tener **rango completo**

Preguntas:

- ¿Cuantas observaciones de landmarks (x, y) se necesitan para resolver la pose de un robot?

Rango

- En SLAM con landmarks, el sistema puede quedar indeterminado
- El rango de $\mathbf{H}$ es **menor o igual** que la suma de los rangos de todas las observaciones
- Para determinar una **solución única**, el sistema debe tener **rango completo**

Preguntas:

- ¿Cuantas observaciones de landmarks (x, y) se necesitan para resolver la pose de un robot?
  - Se necesitan 2 observaciones como mínimo

Rango

- En SLAM con landmarks, el sistema puede quedar indeterminado
- El rango de $\mathbf{H}$ es **menor o igual** que la suma de los rangos de todas las observaciones
- Para determinar una **solución única**, el sistema debe tener **rango completo**

Preguntas:

- ¿Cuantas observaciones de landmarks (x, y) se necesitan para resolver la pose de un robot?
  - Se necesitan 2 observaciones como mínimo
- ¿Cuantas observaciones bearing-only se necesitan para resolver la pose de un robot?

Rango

- En SLAM con landmarks, el sistema puede quedar indeterminado
- El rango de $\mathbf{H}$ es **menor o igual** que la suma de los rangos de todas las observaciones
- Para determinar una **solución única**, el sistema debe tener **rango completo**

Preguntas:

- ¿Cuantas observaciones de landmarks (x, y) se necesitan para resolver la pose de un robot?
  - Se necesitan 2 observaciones como mínimo
- ¿Cuantas observaciones bearing-only se necesitan para resolver la pose de un robot?
  - Se necesitan 3 observaciones como mínimo

Sistema indeterminado

- No hay garantía que un sistema tenga rango completo
  - Los landmarks puede ser observados una sola vez
  - El robot puede que no tenga información de odometría
- Podemos lidiar con estos problemas utilizando un *damping factor* para $\mathbf{H}$
- En vez de resolver $\mathbf{H}\boldsymbol{\Delta}\mathbf{x} = -\mathbf{b}$, resolvemos $(\mathbf{H} + \lambda\mathbf{I})\boldsymbol{\Delta}\mathbf{x} = -\mathbf{b}$

Levenberg–Marquardt

- El damping factor $\lambda\mathbf{I}$ hace que el sistema sea definido positivo
- Es una suma ponderada del método de Gauss-Newton y el método de descenso por el gradiente
- El damping factor regula la convergencia usando acciones de backup/restore

1: **procedure** Levenberg–Marquardt($\mathbf{x}$) ▷ $\mathbf{x}$: semilla inicial
2: **while** (**do**!converged)
3: $\lambda = \lambda_{\text{init}}$
4: $< \mathbf{H}, \mathbf{b} >= buildLinearSystem(\mathbf{x})$
5: $E = error(\mathbf{x})$
6: $\mathbf{x}_{\text{old}} = \mathbf{x}$
7: $\mathbf{\Delta x} = solveSparse((\mathbf{H} + \lambda\mathbf{I})\mathbf{\Delta x} = -\mathbf{b})$
8: $\mathbf{x} \mathrel{+}= \mathbf{\Delta x}$
9: **if** $E < error(\mathbf{x})$ **then**
10: $\mathbf{x} = \mathbf{x}_{\text{old}}$
11: $\lambda \mathrel{*}= 2$
12: **else**
13: $\lambda \mathrel{/}= 2$
14: **end if**
15: **end while**
16: **end procedure**

Material para Bundle Adjustment

Hacer slides de Bundle Adjustment

- Cyrill Stachniss - The Basics about Bundle Adjustment
  Vídeo: `https://youtu.be/sobyKHwgB0Y?si=EqmuOjWNwKBI9jqH`
  Slides: `https://www.ipb.uni-bonn.de/html/teaching/photo12-2021/2021-pho2-09-BA-part1.pptx.pdf`
- Cyrill Stachniss - The Numerics of Bundle Adjustment
  Vídeo: `https://youtu.be/LKDLcKrWOIU?si=InRBlf5Nmf7mGM9H`
  Slides: `https://www.ipb.uni-bonn.de/html/teaching/photo12-2021/2021-pho2-10-BA-part2.pptx.pdf`

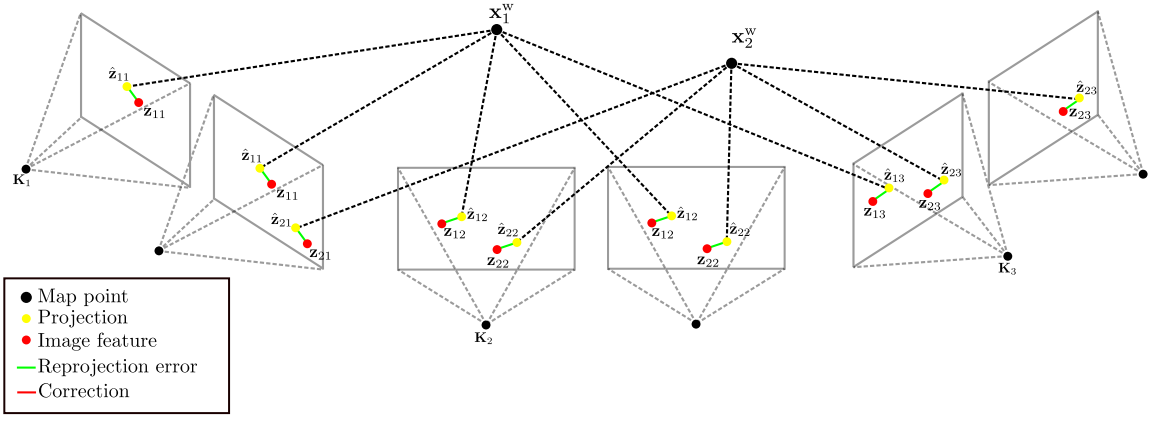
Bundle Adjustment

Bundle Adjustment es un caso

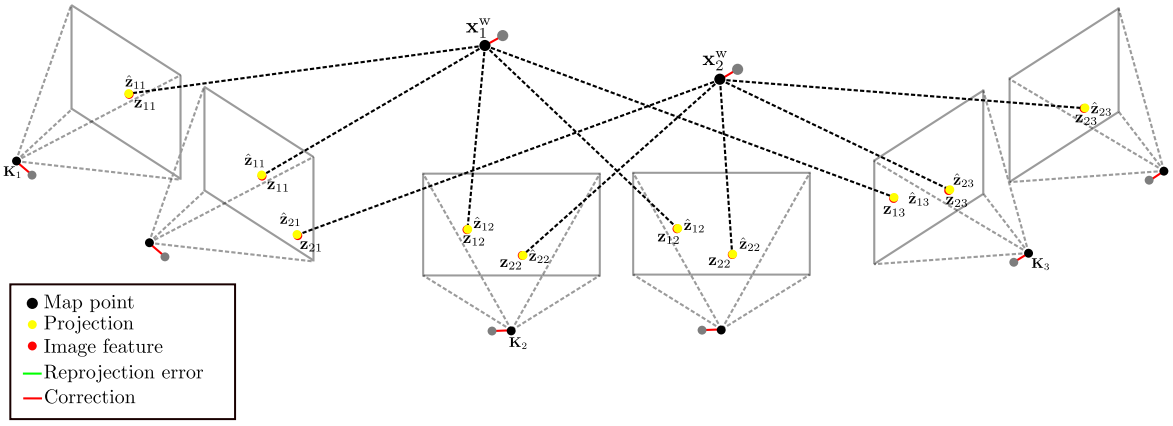
- Reconstrucción 3D basada en imágenes tomadas de diferentes puntos de vista
- Minimiza el error de reproyección en el plano de la imagen 2D
- No hay noción de odometría (pose-pose constraints)
- En general utiliza como método de minimización Levenberg-Marquart
- Desarrollado en el área de Fotogrametría[3] durante la década de 1950

[3] Fotogrametría es la técnica cuyo objeto es estudiar y definir con precisión la forma, dimensiones y posición en el espacio de un objeto cualquiera, utilizando esencialmente medidas hechas sobre una o varias fotografías de ese objeto

Bundle Adjustment



Bundle Adjustment



Bundle Adjustment

Enfoque de mínimos cuadrados para estimar poses de cámara y puntos 3D
Idea Clave:

- Comenzar con una solución inicial (*initial guess*)
- Proyectar los puntos 3D estimados en las imágenes de las cámaras estimadas
- Comparar ubicaciones de las proyecciones de los puntos 3D con los puntos medidos 2D
- Ajustar para minimizar el error en las imágenes

Bundle Adjustment

Agregar contenido de Bundle Adjustment siguiendo el curso de Cyrill Stachniss

Material

Material para armar estas slides:

- Cyrill Stachniss SLAM https://youtu.be/BuRCJ2fegcc
- Cyrill Stachniss Factor Graph https://youtu.be/uuiaqGLFYa4
- Cyrill Stachniss Graph-based SLAM using Pose Graphs https://youtu.be/uHbRKvD8TWg
- Cyrill Stachniss Graph-Based SLAM with Landmarks https://youtu.be/mZBdPgBtrCM
- Wolfram Burgard, Giorgio Grisetti, and Cyrill Stachniss: Graph-based SLAM https://youtu.be/Alu59K8zvYs
- Frank Dellaert https://youtu.be/tm4E1o11kGo
- Frank Dellaert and Michael Kaess - Factor Graphs for Robot Perception https://www.cs.cmu.edu/~kaess/pub/Dellaert17fnt.pdf
- Cyrill Stachniss Slides http://ais.informatik.uni-freiburg.de/teaching/ss13/robotics/slides/16-graph-slam.pdf
- A Technical Walkthrough of the SLAM Back-end https://youtu.be/FhwFyA0NQkE

Más material

Material para armar estas slides:

- Joan Solá - Lie theory for the roboticist `https://youtu.be/gy8U7S4LWzs`
- Joan Solá - Course on SLAM `https://raw.githubusercontent.com/joansola/slamtb/graph/courseSLAM.pdf`
- Michael Kaess - Factor Graphs for Robot Perception `https://youtu.be/Q313pTMAdcM`
- Philipp Hennig - Probabilistic Machine Learning - Lecture 17 - Factor Graphs `https://youtu.be/fXD6KJB1U20`
- Lennart Svensson - Machile Learning Tutorial: Factor Graphs, Belief Propagation and Variational Techniques `https://youtu.be/yQg34-2QhvE`

Bibliografía

- Capítulo 46 de [4]
- Capítulo 10 de [5]

[1] Nived Chebrolu y col. «Adaptive Robust Kernels for Non-Linear Least Squares Problems». En: *(IEEE) Robotics and Automation Letters* 6.2 (2021), págs. 2240-2247. doi: 10.1109/LRA.2021.3061331.

[2] Frank Dellaert y Michael Kaess. 2017, págs. 1-154. doi: 10.1561/2300000043.

[3] Giorgio Grisetti y col. «A Tutorial on Graph-Based SLAM». En: *IEEE Intelligent Transportation Systems Magazine* 2.4 (2010), págs. 31-43. doi: 10.1109/MITS.2010.939925.

[4] Bruno Siciliano y Oussama Khatib. *Springer Handbook of Robotics*. 2nd. Springer Publishing Company, Incorporated, 2016. isbn: 3319325507.

[5] Sebastian Thrun, Wolfram Burgard y Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005. isbn: 0262201623.