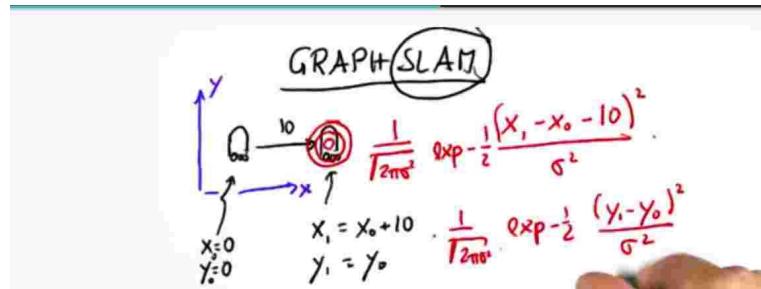


GRAPH SLAM

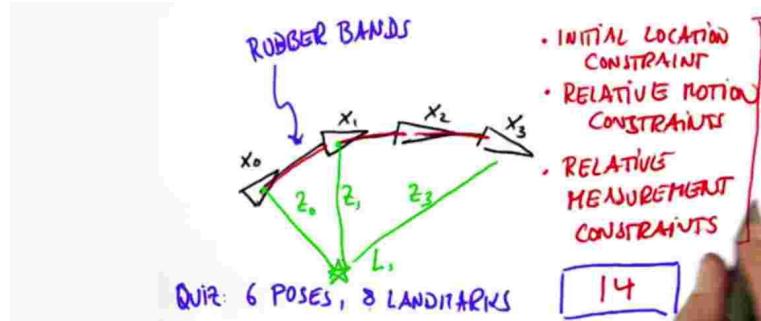
Source: Udacity AI course
Quick notes by P.Phairatt 7.12.16

Basic idea



Assume we have a robot moving and observing the landmarks with a Gaussian distribution. The product of these Gaussian distributions is presented as a constraint in which we want to maximise the likelihood of X1 given X0 (for a motion).

Graph Slam is the approach to define the probability using the sequence of constraints. Following examples demonstrate the method of Graph Slam to update the constraints of each position and landmark in the environment.



Let say, there are 5 motions from X0 and 8 measurements (to L1)

5 motions (result in 6 poses), and 8 landmark measurements = 14 constraints

Information Matrix (Omega) and Vector (Si):

Updating position constraints

	x_0	x_1	x_2	L_0	L_1
x_0	1	-1			
x_1	-1	1			
x_2			1		
L_0				-5	5
L_1				5	-5

$(x_0 \rightarrow x_1) \downarrow 5$
 $x_1 = x_0 + 5$
 $x_0 - x_1 = -5$
 $x_1 - x_0 = 5$
 $x_1 \rightarrow x_2 \quad -4$

from $X_0 \rightarrow X_1$

given $x_1 = x_0 + 5$
 $x_0 - x_1 = -5$
 $-x_0 + x_1 = 5$

therefore

$$\begin{matrix} 1 & -1 \\ -1 & 1 \end{matrix} \quad \begin{matrix} -5 \\ 5 \end{matrix}$$

This is where we are now.

from $X_1 \rightarrow X_2$

given $x_2 = x_1 - 4$

$$\begin{aligned} x_1 - x_2 &= 4 \\ -x_1 + x_2 &= -4 \end{aligned}$$

therefore

$$\begin{array}{ccc} 1 & -1 & 4 \\ -1 & 1 & -4 \end{array}$$

update $\begin{array}{cccc} X_0 & X_1 & X_2 & Si \\ X_0 & 1 & -1 & -5 \\ X_1 & -1 & 1+1 & 5+4 \\ X_2 & -1 & 1 & -4 \end{array}$

Update landmark constraints

between X_1 and L_0

given landmark $L_0 = x_1 + 9$

$$\begin{aligned} x_1 - L_0 &= -9 \\ -x_1 + L_0 &= 9 \end{aligned}$$

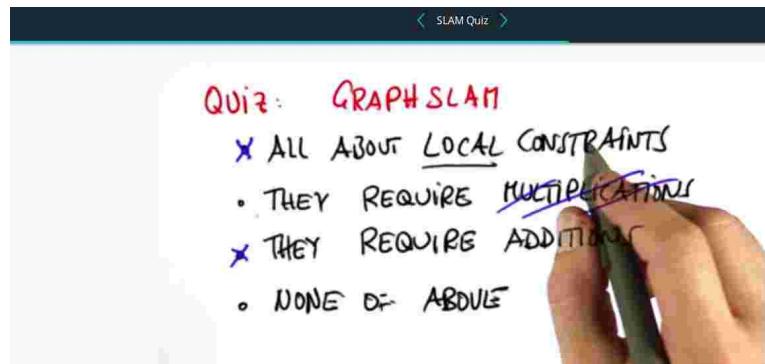
therefore,

$$\begin{array}{ccc} 1 & -1 & -9 \\ -1 & 1 & 9 \end{array}$$

update $\begin{array}{ccccc} X_0 & X_1 & X_2 & L_0 & Si \\ X_0 & 1 & -1 & & -5 \\ X_1 & -1 & 2+1 & -1 & -1 & 9-9 \\ X_2 & -1 & 1 & & -4 \\ L_0 & -1 & & 1 & 9 \end{array}$

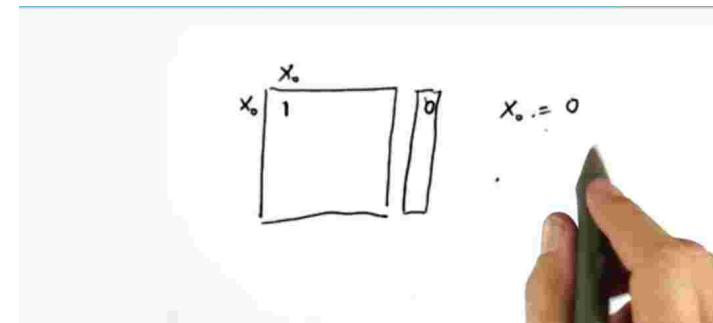
This information matrix and vector will later be called as Omega and Si.

Graph Slam manipulation



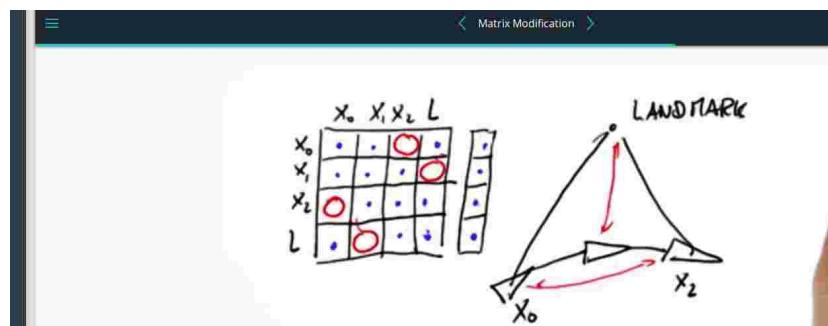
Graph Slam manipulation is just a direct local constraint between motions/landmarks and additions

Initial constraint update



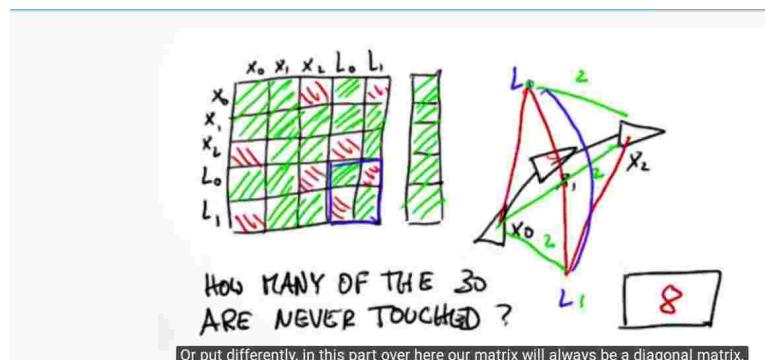
Initial constraint at X_0 , given initial $X_0 = 0$, the information matrix and vector will be initialised with 1 and 0.

No constraint relationship



Red lines show the no direct constraint among the positions and the landmark in Graph Slam that result in no update in the matrix and vector (0).

No constraint relationship: more example



red/black lines = constrained lines

green/blue lines = no constrained lines

1 pairs = 2 constraints missing

There always no constraints between landmarks (e.g. L0 ↔ L1). It is therefore always a diagonal matrix.

Graph Slam Formula

A hand-drawn diagram showing the formula $M = \Omega^{-1} \cdot \xi$. On the left, there is a vertical rectangle labeled M . To its right is an equals sign. To the right of the equals sign is a large square labeled Ω^{-1} . To the right of Ω^{-1} is a vertical rectangle labeled ξ . Below this equation is a large oval containing the same formula $M = \Omega^{-1} \cdot \xi$. A small green arrow points from the oval towards the formula. At the bottom of the oval, the text "and out comes the best places for your robot." is written.

Information matrix (Omega), Information vector (Si), State (u - position and landmark positions)

Handwritten notes showing initial positions and constraints. A 3x3 matrix X_0 is shown with columns x_0, x_1, x_2 and rows x_0, x_1, x_2 . Below it, three equations define x_0, x_1, x_2 in terms of $x_0 + 5$ and $x_1 + 3$. To the right, the formula $\xi = \Omega^{-1} \cdot \xi$ is written with values -3, 2, 5. At the bottom, the note "x1 becomes 2, and x2 becomes 5." is written.

Given initial position and constraints. The equation above should give the state $u = [-3, 2, 5]$

Handwritten notes showing the update of the information matrix and vector. It shows the addition of three matrices (1x1, 1x1, 1x1) to form a 3x3 matrix Ω , and the addition of three vectors (-3, 2, 5) to form a vector ξ .

Update motions accordingly,

```
Omega: [2.000, -1.000, 0.000]
Omega: [-1.000, 2.000, -1.000]
Omega: [0.000, -1.000, 1.000]

Xi:   [-8.000]
Xi:   [2.000]
Xi:   [3.000]

Result: [-3.000]
Result: [2.000]
Result: [5.000]
```

There is an empty function in your code that accepts as parameter

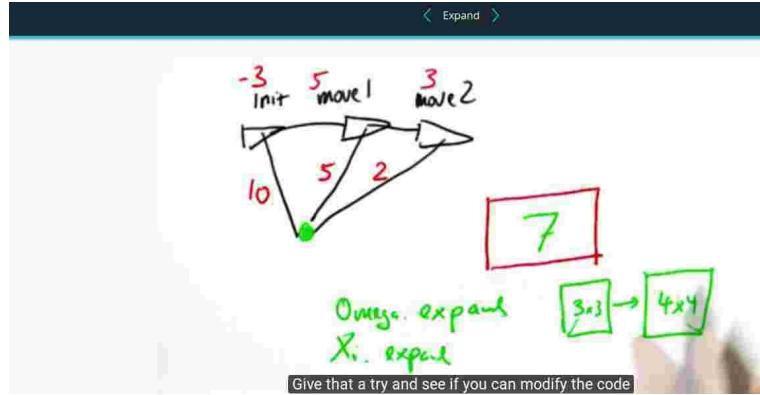
$u = \text{Omega_Inverse} * \text{Si} = [-3, 2, 5]$

Example with 1 landmark constraint and no noise

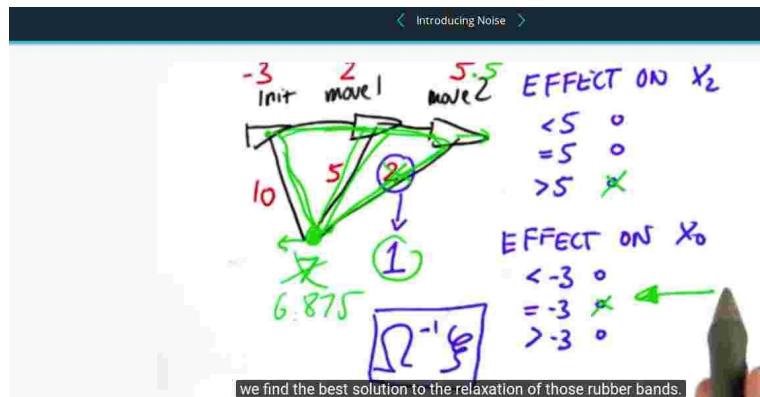
It is clear that the $L_0 = 7$ (with no measurement noise).

$$X_1 = X_0 + 5, X_2 = X_1 + 3$$

$$L_0 = X_0 + 10, X_1 + 5, X_2 + 2 \text{ where } X_0 = -3, X_1 = 2, X_2 = 5$$



Example with noise



Let add noise to the last measurement constrained to X_2 ! Say the measurement is now 1 rather than 2 (noise free case). What will happen to L_0 , X_0 and X_1 (comparing with noise free case)?

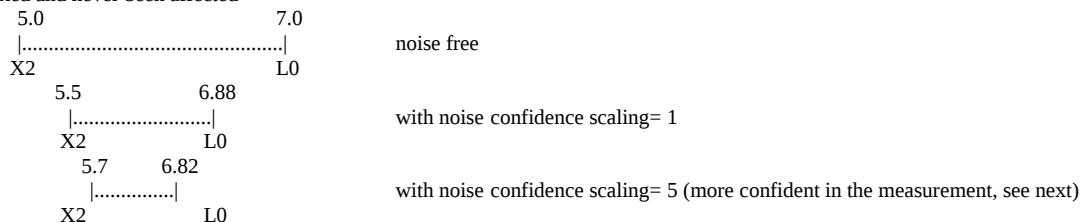
Imagine the rubber band between $X_2 \leftrightarrow L_0$, it has been pull together tighter by the measurement as the result

- the 'distance' between X_2 and L_0 will be reduced (pull closer) by '**increasing X_2 and reducing L_0 '**

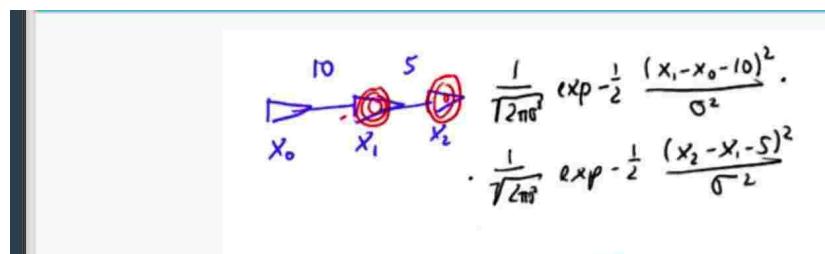
($L_0 - X_2 = 2$ noise free, $L_0 - X_2 = 1$ with noise)

- X_1 has been pull up as well to maximise the likelihood

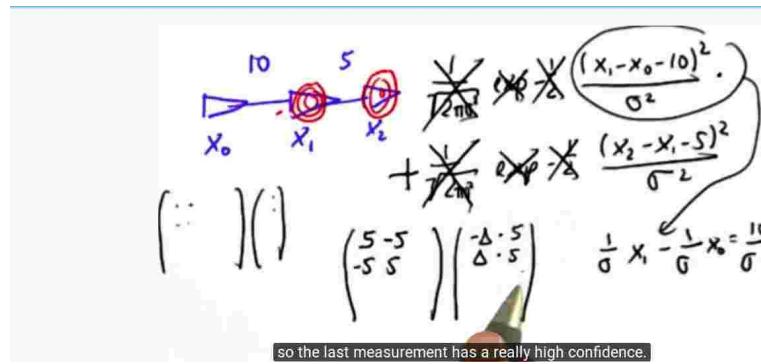
- X_0 is fixed and never been affected



Confident Measurement

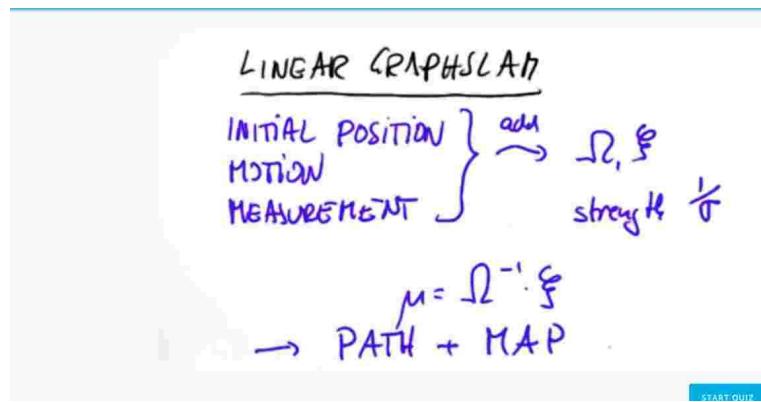


Total probability of the system is the product of these two constraints that we want to maximise.



The maximisation trick is the sum of these term scaling with the confidence level as depicted in the picture above in one motion step. In the Graph Slam, it is simply multiples with the scaling number. Considering only one term (between X_0 and X_1), the scaling factor is proportional to the variance (low variance, high confidence \rightarrow larger scaling factor)

Put it all together



Given initial position, then motion, then landmark update for each navigation with confident strength, so we can build the estimate path the robot takes.

Note on the Graph Slam approach

Method 1- Separate X, Y vector (S_i), sharing the same matrix (Ω)

Method 2- Combine X, Y state

full omega matrix dim = $2*(N \text{ motion} + m \text{ landmarks}) \rightarrow \text{motion include initial, } 2x \text{ for X, Y Coordinate}$

Separate- matrix $N+M$

P0 P1 P2 L0 L1

P0

P1

P2

OMEGA

L0

L1

X0

X1

X2

SI_X

Lx0

Lx1

Y0

Y1

Y2

SI_Y

Ly0

Ly1

Combine- matrix $2*(N+M)$

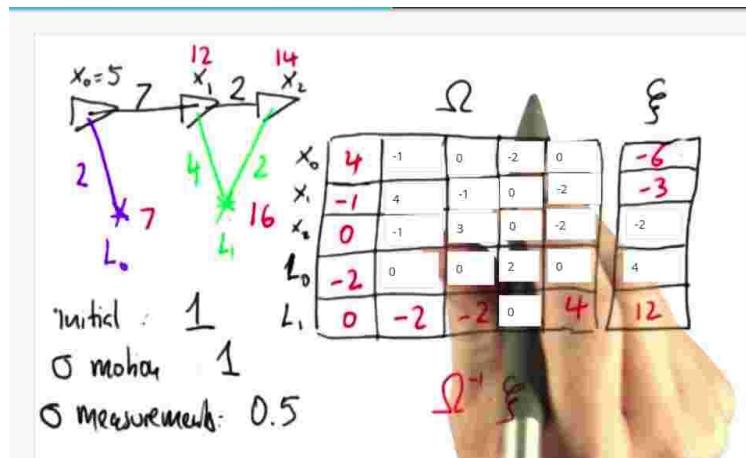
X0	Y0	X1	Y1	X2	Y2	Lx0	Ly0	Lx1	Ly1
X0									
Y0									
X1									
Y1									
X2									
Y2									
Lx0									
Ly0									
Lx1									
Ly1									

X0	Y0	X1	Y1	X2	Y2	Lx0	Ly0	Lx1	Ly1
X0									
Y0									
X1									
Y1									
X2									
Y2									
Lx0									
Ly0									
Lx1									
Ly1									

OMEGA

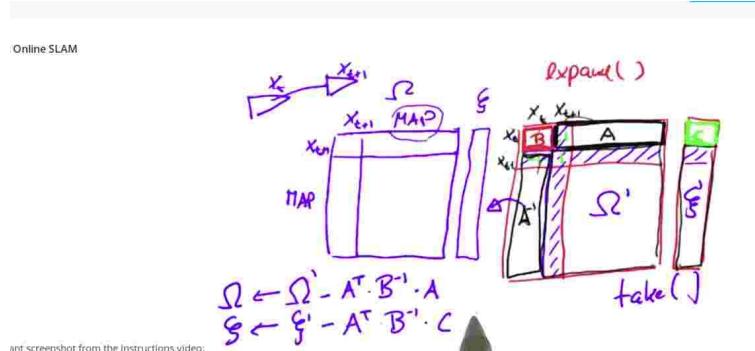
SI

QUIZ Filling full Graph Slam matrix



```
# Motion constraints
X1 = X0 + 7
X2 = X1 + 2
# Landmark constraints
L0 = X0 + 2
L1 = X1 + 4
L1 = X2 + 2
```

Online GraphSLAM- to make it scalable



Scaling to the recent position by only keeping track of the recent robot position after the motion and measurement update.
Add the motion/measurement update as the full slam, then summarise to the recent state by the above formula.