Graph based SLAM

Seungwon Choi

2020.02.17

Contents

1. Least Square Method

1.1 Vector Projection and Projection Matrix

Figure 1의 왼쪽 그림과 같이 임의의 두 벡터 $\mathbf{a}, \mathbf{b}$ 가 주어졌을 때, $\mathbf{p}$를 $\mathbf{a}$에 대한 $\mathbf{b}$의 투영 이라고 정의한다.

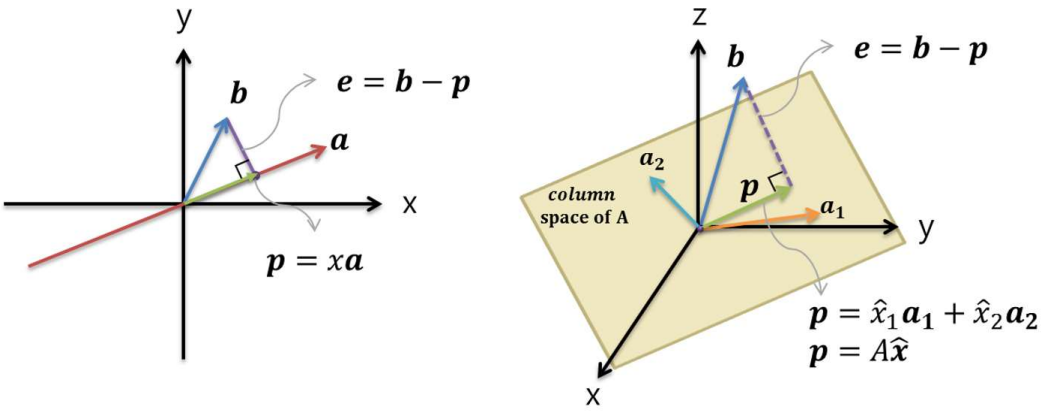

Figure 1 (좌) 2D Vector Projection (우) 3D Vector Projection

이 때 $\mathbf{p}$와 $\mathbf{b}$의 차이를 $\mathbf{e}$로 정의하며 $\mathbf{e}$와 $\mathbf{p}$는 서로 수직(perpendicular)이기 때문에, 아래의 수식(1)이 성립함을 알 수 있다..

$$\mathbf{a}^{\mathrm{T}}(\mathbf{b} - \mathrm{x}\mathbf{a}) = 0 \tag{1}$$

위 수식(1)을 전개하면 $\mathrm{x} = \frac{\mathbf{a}^{\mathrm{T}}\mathbf{b}}{\mathbf{a}^{\mathrm{T}}\mathbf{a}}$ 의 결과를 얻을 수 있고, P를 $\frac{\mathbf{a}\mathbf{a}^{\mathrm{T}}}{\mathbf{a}^{\mathrm{T}}\mathbf{a}}$ 로 정의하면 수식(2)가 성립한다.

이때 P를 투영 행렬(Projection Matrix)로 정의한다.

$$\mathbf{p} = \mathrm{P}\mathbf{b} \tag{2}$$

1.2 Geometrical Meaning of Vector Projection

임의의 행렬 A에 대하여, A의 일차 독립인 열 벡터(column vector)들의 선형 조합(linear combination)으로 만들어 지는 공간을 column space라고 한다. 이 행렬에 임의의 벡터 $\mathbf{x}$를 곱한다는 것은 수식(3)과 같이 $\mathbf{x}$를 A의 column space로 안착(landing) 하는 것을 의미한다. 즉 행렬 A와 벡터 $\mathbf{x}$의 연산의 결과는 행렬 A의 column space내부에 존재하는 벡터임을 의미한다.

$$\begin{matrix} \mathrm{A} & \mathbf{x} \\ \begin{bmatrix} 1 & 1 \\ 2 & 3 \end{bmatrix} \end{matrix} \begin{bmatrix} 4 \\ 7 \end{bmatrix} = \begin{matrix} 4 * \mathbf{col_1} \\ 4\begin{bmatrix} 1 \\ 2 \end{bmatrix} \end{matrix} + \begin{matrix} 7 * \mathbf{col_2} \\ 7\begin{bmatrix} 1 \\ 3 \end{bmatrix} \end{matrix} = \begin{matrix} \mathbf{b} \\ \begin{bmatrix} 11 \\ 29 \end{bmatrix} \end{matrix} \tag{3}$$

이는 행렬 A의 column space가 해당 행렬의 column vector의 모든 가능한 선형 조합의 집합이기

때문이다.

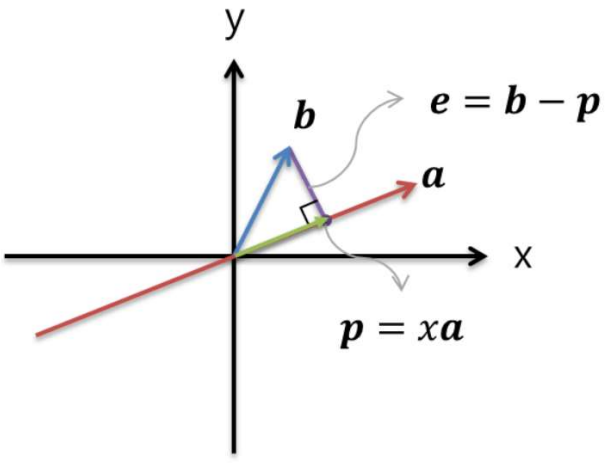

Figure 2 2D Vector Projection

같은 논리에서 식(2)의 투영 행렬 P의 column space는 Figure 2의 벡터 $\mathbf{a}$를 지나는 직선이 된다. 이는 임의의 벡터 $\mathbf{b}$가 투영 행렬 P와의 연산을 통해 $\mathbf{a}$를 지나는 직선 위에 안착(landing)하기 때문이다.

투영 행렬의 다른 성질은 rank가 1인 symmetric 행렬 이라는 것 이다. P는 $\mathbf{aa^T}/\mathbf{a^Ta}$로 정의되고, $\mathbf{aa^T}$는 같은 벡터를 각각 column과 row로 내적 한 결과이므로 rank가 1인 symmetric한 행렬이 된다. 또한, 같은 이유로 벡터 $\mathbf{a}$는 투영 행렬 P의 기저(basis)가 된다.

1.3 Approximation using Vector Projection

Overdetermined System 이란 미지수의 개수보다 식의 수가 더 많은 경우를 의미한다. 따라서 이 경우에는 해가 존재하지 않으며, 해당 System을 풀기 위해서는 일반적으로 근사 해를 구한다. Overdetermined System이 해가 존재하지 않는 이유를 column space의 관점에서 보면, 수식(4)의 벡터 $\mathbf{b}$가 행렬 A의 column space 외부에 존재하기 때문으로 해석할 수 있다.

$$\mathbf{Ax} = \mathbf{b} \tag{4}$$

따라서 사실상 이는 수식(4)의 두 변이 일치하지 않는다는 것을 의미한다.

A는 이미 정해진 System 이기 때문에 해당 시스템의 근사 해($\hat{\mathbf{x}}$)를 구하기 위해서는 결국 $\mathbf{b}$의 값을 바꾸어야 한다. 이 중에 가장 적절한 근사 해를 찾기 위해서는 A의 column space 내부의 벡터 중 $\mathbf{b}$와 가장 유사한 벡터를 이용해야 한다. 이 때 벡터 투영의 개념이 사용된다. 벡터 $\mathbf{b}$의 A의 column space로의 투영이 해당 space내에서 $\mathbf{b}$와 가장 유사한(Figure 3에서 e가 가장 작은) 벡터이기 때문이다. 이 벡터를 $\mathbf{p}$라고 하면, 근사 해를 찾는다는 것을 결국 수식(5)를 푼다는 것과 같은 의미로 해석할 수 있다.

$$\mathrm{A}\hat{\mathbf{x}} = \mathbf{p} \tag{5}$$

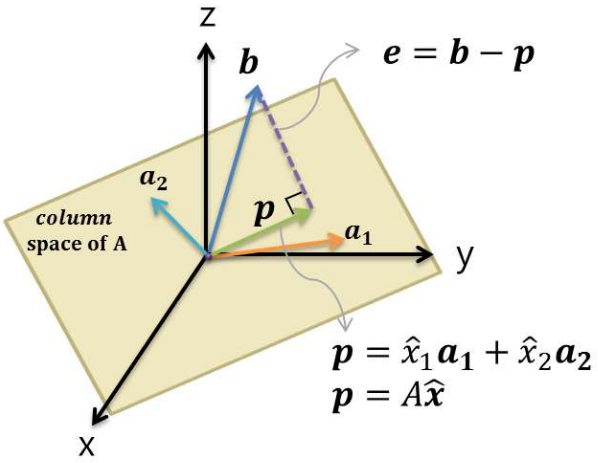

Figure 3 3D Vector Projection

이 과정을 정리하면 수식(6)~(10)과 같고, 수식(10)과 같은 방식으로 근사 해를 구하는 것을 Least Square Method라고 한다.

$$\mathbf{b} - \mathrm{A}\hat{\mathbf{x}} \perp \text{plane} \tag{6}$$

$$\begin{bmatrix} \mathbf{a_1^T} \\ \mathbf{a_2^T} \end{bmatrix} (\mathbf{b} - \mathrm{A}\hat{\mathbf{x}}) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{7}$$

$$\mathrm{A^T}(\mathbf{b} - \mathrm{A}\hat{\mathbf{x}}) = \mathbf{0} \tag{8}$$

$$\mathrm{A^T A}\hat{\mathbf{x}} = \mathrm{A^T}\mathbf{b} \tag{9}$$

$$\hat{\mathbf{x}} = (\mathrm{A^T A})^{-1}\mathrm{A^T}\mathbf{b} \tag{10}$$

Figure 4와 같이 3개의 데이터 (1,1), (2,2), (3,2)를 얻었을 때, 이 데이터를 가장 잘 설명할 수 있는 직선을 구하는 문제를 가정해보자. 여기서, 모든 데이터를 관통하는 직선을 구할 수는 없기 때문에 추정 값과 실제 데이터 값의 차이를 최소화 하는 직선을 찾고자 하는 것이다.

$$\mathrm{b} = \mathrm{ct} + \mathrm{d} \tag{11}$$

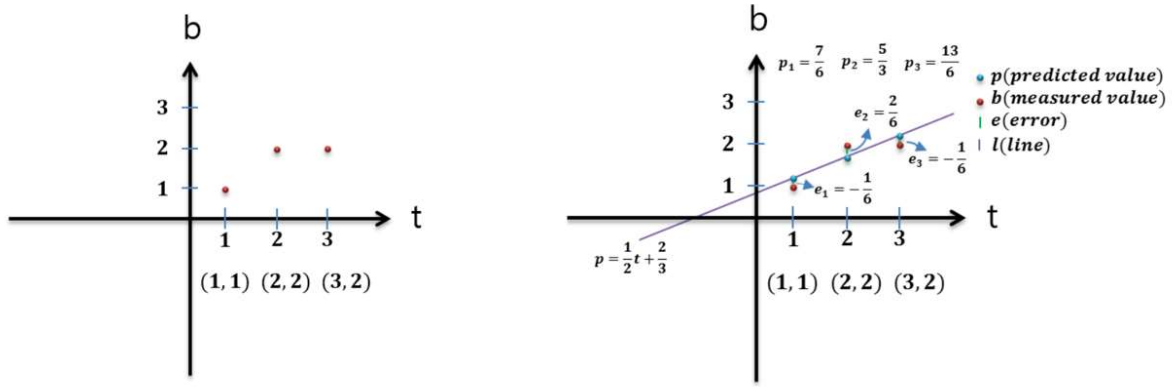

Figure 4 (좌)데이터의 예 (우)Least Square Method를 통해 얻은 직선

해당 직선을 수식(11)과 같이 표현하면 미지수는 c와 d 이므로 2개이고, 얻어진 데이터는 총 3개이므로 이는 Overdetermined System에 해당한다. 이를 행렬 식으로 나타내면 수식(12)와 같다. 이를 수식(10)을 이용하면 Figure 4의 오른쪽 그림과 같은 직선을 얻을 수 있다.

$$\begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} c \\ d \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix} \tag{12}$$

1.4 Non-linear least squares – Gauss-Newton Method

앞서 설명한 Least Square Method는 찾고자 하는 System이 선형인 경우에만 적용 가능하다. 비선형 Least square 문제에서는 이 절에서 설명할 Gauss-Newton 방법을 적용하면 해당 파라미터(수식(12)에서 c,d와 같은)를 추정할 수 있다.

이를 위해서는 먼저 뉴턴법(Newton's Method)에 대한 이해가 선행되어야 한다. 이 방법은 방정식 $f(x) = 0$의 해를 근사적으로 찾을 때 유용하게 사용되는 방법이다. 예를 들어 아래와 같은 복잡한 함수가 있다고 가정해 보자.

$$f(x) = x^7 - 2x^6 + x^5 + 7x^2 - 3x + 11 = 0$$

위 식은 인수분해가 되지 않기 때문에, 해당 식의 해를 구하는 것은 해석적인(Analytic)방법으로는 어렵다는 것을 알 수 있다.

뉴턴법은 미분 계수 $f'(x_1)$가 $x = x_1$ 에서의 접선의 기울기라는 미분의 기하학적 해석에 기초한다. Figure 5와 같이, 일단은 아무값이나 $x = x_1$를 넣고 $f(x_1)$의 값을 살펴본다. 만일 $f(x_1) > 0$이고 $f'(x_1) > 0$라면 $f(x) = 0$이 되는 x는x_1보다 작은 x_2에 더 근접한 값일 것 이라는 논리이다.

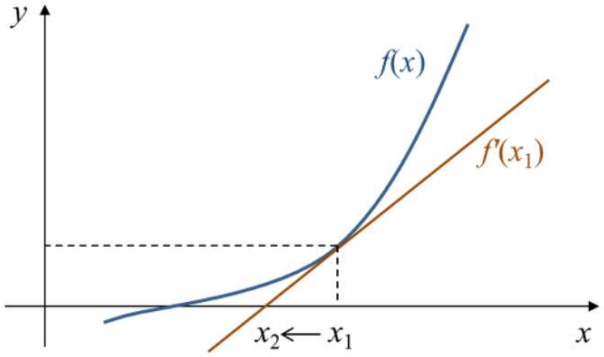

Figure 5 Newton Method의 예

연립 방정식의 근사해를 구할 때는 보통 Gauss-Newton Method 를 활용한다. 이는 Newton Method 를 연립 방정식으로 확장한 것으로 볼 수 있으며, 일차 미분은 행렬에 대한 Jacobian 으로 표현된다.

1.5 Least Square Method in SLAM

SLAM에서는 센서로부터 측정된 measurement에 가장 적합한 로봇의 state를 계산한다. 로봇의 특정 시점에서의 상태를 알고 있을 때, 해당 상태에서의 예상되는 measurement($\hat{z}_i$) 와 실제 measurement(z_i)의 차이를 Least Square Method로 최소화 함으로써 이를 달성한다. 이는 SLAM 문제를 Figure 5와 같은 System을 푸는 문제로 이해할 때, 즉 관측되는 센서 데이터 각각이 하나의 방정식이 될 때, 구하고자 하는 해 state(unknown)의 수가 주어진 방정식의 수보다 적은 Overdetermined System 이기 때문이다.

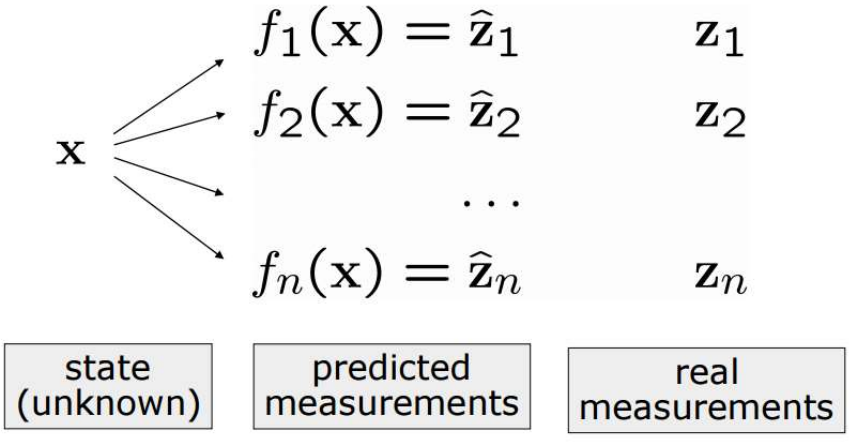

Figure 6 SLAM에서의 unknown과 센서 데이터를 통해 얻어지는 equations

Error($\mathbf{e}_i$)는 실제 measurement(z_i)와 특정 시점의 상태에서 예상되는 measurement($\hat{z}_i$)의 차이를 나타내며, 수식(13)과 같이 표현된다.

$$e_i(\mathbf{x}) = \mathbf{z_i} - f_i(\mathbf{x}) \tag{13}$$

measurement의 noise는 평균이 0인 Gaussian 분포를 따른다고 가정하고 measurement의 covariance matrix의 inverse를 Information matrix(Ω_i)라고 정의하면, MLE를 통해 최소화 시켜야 할 오차는 수식(14)와 같다.

$$e_{\mathrm{squared,i}} = e_i^T(\mathbf{x})\Omega_i e_i(\mathbf{x}) \tag{14}$$

즉 로봇의 최적화된 state를 계산하는 문제는, 수식(15)와 같이 주어진 모든 센서 measurement의 오차가 최소가 되는 state를 찾는 문제이다.

$$\mathbf{x}^* = \mathrm{argmin}_{\mathbf{x}} \sum_i e_i^T(\mathbf{x})\Omega_i e_i(\mathbf{x}) \tag{15}$$

수식(15)는 Gaussian 분포와 수식(16)에 표현된 Bayes filter의 posterior로부터 유도된다.

$$p(x_{0:t}|z_{1:t}, u_{1:t}) = \eta p(x_0) \prod_t [p(x_t|x_{t-1}, u_t)p(z_t|x_t)] \tag{16}$$

$$\log[p(x_{0:t}|z_{1:t}, u_{1:t})] = \mathrm{const} + \log[p(x_0)] + \sum_t [\log[p(x_t|x_{t-1}, u_t)] + \log[p(z_t|x_t)]] \tag{17}$$

$$\text{Assume : all probability} \sim N\left(x, \mu, \Sigma\right) \text{ (bayes filter theory)}$$

$$\log N\left(x, \mu, \Sigma\right) = \mathrm{const} - \frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu) \tag{18}$$

$$\log[p(x_{0:t}|z_{1:t}, u_{1:t})] = \mathrm{const} - \frac{1}{2}e_p(x) - \frac{1}{2}\sum_t \left[e_{u_t}(x) + e_{z_t}(x)\right] \tag{19}$$

따라서 posterior의 최댓값은 찾는 것은 error function $e_p(x) + \sum_t \left[e_{u_t}(x) + e_{z_t}(x)\right]$ 을 최소화 하는 것과 같다.

정리하자면, 수식(16)~(19)에서의 유도로 SLAM에서 풀고자 하는 문제는 결국 아래의 수식(20)을 만족하는 state, 즉 현재 시점에서 관측된 센서의 측정 결과와 가장 잘 맞는 state를 찾는 문제로 귀결된다.

$$\mathbf{x}^* = \text{argmin}_{\mathbf{x}} \sum_i e_i^T(\mathbf{x}) \Omega_i e_i(\mathbf{x}) \tag{20}$$

수식(20)의 해는 해당 에러 값을 0으로 만드는 값이다. 하지만, 일반적으로 복잡한 에러 식에서 해당 해가 존재하기 어렵기 때문에, 일반적으로 수치적인 접근 방법인 Gauss-Newton Method를 활용한다.

수치적인 접근 방법에서 에러 함수는 local minima 근처에서 smooth 하다는 가정이 전제된다. 이러한 전제 하에, current solution을 기준으로 1차 Taylor 근사 하여 에러를 1차식으로 만든 다(선형화) 그런 다음 이를 제곱하여 Convex 형태로 만들고, 이 Squared 에러의 1차 미분이 0이 되는 Solution을 반복해서 구하면 Global minima에 가까워질 수 있다.

이러한 방식으로 로봇의 다음 state를 구하는 것은 수식(21)~(30)에 표현되어 있다.

$$e_i(\mathbf{x} + \Delta\mathbf{x}) \approx e_i(\mathbf{x}) + \frac{\partial e_i(\mathbf{x})}{\partial \mathbf{x}} \Delta\mathbf{x} \approx e_i(\mathbf{x}) + J_i(\mathbf{x})\Delta\mathbf{x} \tag{21}$$

$$e_{\text{squared},i}(\mathbf{x} + \Delta\mathbf{x}) = e_i^T(\mathbf{x} + \Delta\mathbf{x}) \Omega_i e_i(\mathbf{x} + \Delta\mathbf{x}) \tag{22}$$

$$e_{\text{squared},i}(\mathbf{x} + \Delta\mathbf{x}) \approx (e_i(\mathbf{x}) + J_i(\mathbf{x})\Delta\mathbf{x})^T \Omega_i (e_i(\mathbf{x}) + J_i(\mathbf{x})\Delta\mathbf{x}) \tag{23}$$

$$\text{Notation}: \quad e_i(x) = e_i, \qquad J_i(x) = J_i$$

$$e_{\text{squared},i}(\mathbf{x} + \Delta\mathbf{x}) \approx e_i^T \Omega_i e_i + e_i^T \Omega_i J_i \Delta\mathbf{x} + \Delta\mathbf{x}^T \mathbf{J_i^T} \Omega_i e_i + \Delta\mathbf{x}^T J_i^T \Omega_i J_i \Delta\mathbf{x} \tag{24}$$

$$e_{\text{squared},i}(\mathbf{x} + \Delta\mathbf{x}) \approx e_i^T \Omega_i e_i + 2 e_i^T \Omega_i J_i \Delta\mathbf{x} + \Delta\mathbf{x}^T J_i^T \Omega_i J_i \Delta\mathbf{x} \quad (\because \mathbf{scalar^T} = \mathbf{scalar}) \tag{25}$$

$$\text{Notation}: e_i^T \Omega_i e_i = c_i, \qquad e_i^T \Omega_i J_i = b_i^T, \qquad J_i^T \Omega_i J_i = H_i$$

$$e_{\text{squared},i}(\mathbf{x} + \Delta\mathbf{x}) = c_i + 2 b_i^T \Delta\mathbf{x} + \Delta x^T H_i \Delta\mathbf{x} \tag{26}$$

$$F(\mathbf{x} + \Delta\mathbf{x}) \approx \sum_i (c_i + b_i^T \Delta\mathbf{x} + \Delta\mathbf{x}^T H_i \Delta\mathbf{x}) \tag{27}$$

$$\text{Notation}: \sum_i c_i = c, \qquad \sum_i b_i^T = b^T, \qquad \sum_i H_i = H$$

$$F(\mathbf{x} + \Delta\mathbf{x}) \approx c + 2 b^T \Delta\mathbf{x} + \Delta\mathbf{x}^T H \Delta\mathbf{x} \tag{28}$$

$$\frac{\partial \mathrm{F}(\mathbf{x}+\Delta\mathbf{x})}{\partial \Delta\mathbf{x}} \approx 2\mathrm{b} + 2\mathrm{H}\Delta\mathbf{x} \tag{29}$$

$$\text{If set}: 2\mathrm{b} + 2\mathrm{H}\Delta\mathbf{x} = 0, \qquad \Delta\mathbf{x}^* = -\mathrm{H}^{-1}\mathrm{b}$$

$$\mathbf{x_{next}} = \mathbf{x} + \Delta\mathbf{x}^* \quad \text{(state update)} \tag{30}$$

2. GraphSLAM

2.1 Two Main SLAM Approaches

SLAM에는 두가지 접근 방식이 있다. Online SLAM과 Full SLAM이 그것이다. Online SLAM은 Filter version의 SLAM 방식으로, 식(31)의 확률을 최대화 하는 문제이다.

$$p(x_t|z_{1:t}, u_{1:t}) \tag{31}$$

Online SLAM은 새로운 정보(control 및 measurement)를 얻을 때 마다 수행하며, Markov 가정을 기반으로 현재 최고 추정치(belief)를 생성하고, 이것이 최선이라는 가정 하에 다음 단계에서의 문제에 해당 솔루션을 사용하는 방식이다. EKF SLAM과 FastSLAM, Occupancy Grid SLAM 등이 이에 해당한다.

Full SLAM은 Smoothing version의 SLAM이다. 이 SLAM 방식은 식(32)의 확률을 최대화 하는 문제이다.

$$p(x_{0:t}|z_{1:t}, u_{1:t}) \tag{32}$$

모든 정보(state, measurement, control 등)을 저장 해 놓고, 로봇의 pose와 map을 필요할 때만 계산한다. 일반적으로 Online SLAM보다 더 강건하다.

2.2 Notation and Model

Full SLAM에서는 아래와 같은 Notation을 사용한다.

$$(1)x_t^r = \begin{bmatrix} X_t \\ Y_t \\ \theta_t \end{bmatrix}, (2)m_i = \begin{bmatrix} m_{i,X} \\ m_{i,Y} \end{bmatrix}, (3)m = \begin{bmatrix} m_1 \\ \vdots \\ m_M \end{bmatrix}, (4)x_{0:t} = \begin{bmatrix} x_0^r \\ x_1^r \\ \vdots \\ x_t^r \\ m \end{bmatrix}, (5)x_t = \begin{bmatrix} x_t^r \\ m \end{bmatrix} \tag{33}$$

이는 순서대로 (1)시각 t에서의 로봇의 state, (2)i번째 feature의 좌표, (3)feature map, (4)Full state, (5)시각 t에서의 Full state에 해당한다.

Full SLAM에서의 Motion Model과 Measurement Model은 각각 식(34)와 (35)로 표현된다.

$$x_t^r = g(x_{t-1}^r, u_t) + \delta_t \tag{34}$$

$$z_t = \hbar(x_t^r, m) + \varepsilon_t \tag{35}$$

2.3 Graph and Constraints

GraphSLAM에서는 로봇의 state와 map의 feature를 Node로, state와 state 그리고 state와 map의 feature 사이의 constraint를 Edge로 표현한다.

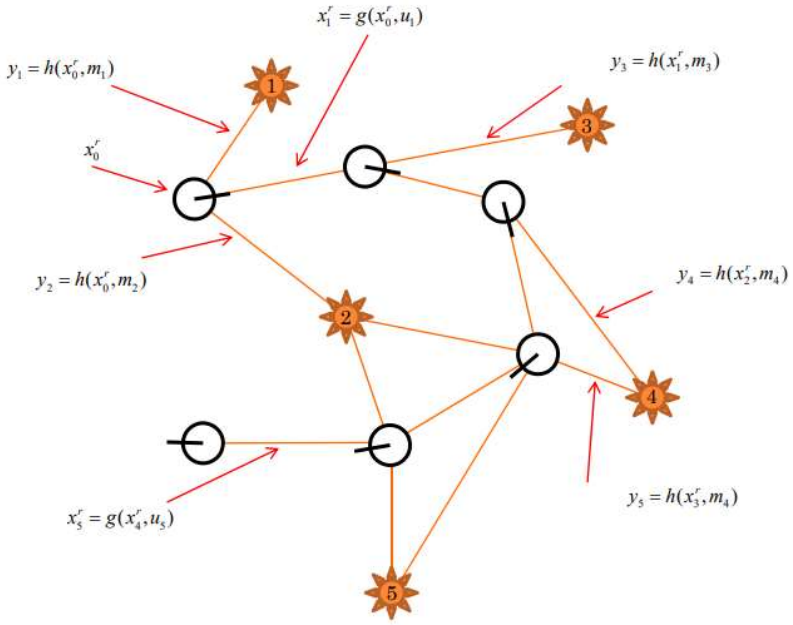

Figure 7 Graph 표현

Constraint란 해당 edge의 불확실성의 척도이며, 이 값을 최소화 하는 것이 GraphSLAM에서 풀어야 할 문제이다.

2.4 GraphSLAM Optimization

GraphSLAM에서는 식(36)의 문제를 풀어야 한다.

$$\max_{x_{0:t}} p(x_{0:t}|z_{1:t}, u_{1:t}) \tag{36}$$

식(36)에 Bayes 정리를 적용하면 식(37)과 같이 표현할 수 있다.

$$\begin{aligned} p(x_{0:t}|z_{1:t}, u_{1:t}) &= \eta p(z_t|x_{0:t}, u_{1:t}) p(x_{0:t}|z_{1:t-1}, u_{1:t}) \\ &= \eta p(z_t|x_t) p(x_{0:t}|z_{1:t-1}, u_{1:t}) \end{aligned} \tag{37}$$

또한 식(37)의 $p(x_{0:t}|z_{1:t-1}, u_{1:t})$ 는 아래의 식(38)와 같이 변형할 수 있다.

$$\begin{aligned} p(x_{0:t}|z_{1:t-1}, u_{1:t}) &= p(x_t^r|x_{0:t-1}, u_{1:t}) p(x_{0:t-1}|z_{1:t-1}, u_{1:t-1}) \\ &= p(x_t^r|x_{t-1}, u_t) p(x_{0:t-1}|z_{1:t-1}, u_{1:t-1}) \\ &= p(x_t^r|x_{t-1}^r, u_t)[\eta \cdot p(z_{t-1}|x_{t-1}) p(x_{0:t-2}|z_{1:t-2}, u_{1:t-2})] \end{aligned} \tag{38}$$

해당 과정을 반복하게 되면, 최종적으로 식(39)를 얻게 된다.

$$p(x_{0:t}|z_{1:t},u_{1:t}) = \eta p(x_0)\prod_{\tau=1}^{t} p(z_\tau|x_\tau)p(x_\tau^r|x_{\tau-1}^r,u_\tau) \\ = \eta p(x_0)\prod_{\tau=1}^{t}\left[\left\{\prod_i p(z_\tau^i|x_\tau)\right\}p(x_\tau^r|x_{\tau-1}^r,u_\tau)\right] \tag{39}$$

이 때, map에 대한 prior가 없다면, $p(x_0)$ 을 $p(x_0^r)$ 로 대체한다.

식(39)를 통해서, 식(36)의 최적화 문제를 식(40)과 같이 표현할 수 있다.

$$\max_{x_{0:t}} p(x_{0:t}|z_{1:t},u_{1:t}) = \max_{x_{0:t}}\left[\eta p(x_0)\prod_{\tau=1}^{t}\left[\left\{\prod_i p(z_\tau^i|x_\tau)\right\}p(x_\tau^r|x_{\tau-1}^r,u_\tau)\right]\right] \\ = \min_{x_{0:t}}\left[-\ln\left(\eta p(x_0)\prod_{\tau=1}^{t}\left[\left\{\prod_i p(z_\tau^i|x_\tau)\right\}p(x_\tau^r|x_{\tau-1}^r,u_\tau)\right]\right)\right] \\ = \min_{x_{0:t}}\left[const. - \ln(p_{x_0}) - \sum_{\tau=1}^{t}\left(\ln\left(p(x_\tau^r|x_{\tau-1}^r,u_\tau)\right)\right) - \sum_{\tau=1}^{t}\sum_i \ln\left(p(z_t^i|x_t)\right)\right] \tag{40}$$

2.5 Gaussian Noise and Optimization

Motion Model과 Measurement Model에는 Noise가 존재하며 이는 Gaussian Noise로 가정한다. 따라서 아래의 수식(41~43)과 같이 해당 모델과 prior를 Gaussian 분포로 표현할 수 있다.

$$p(x_t^r|x_{t-1}^r,u_t) = \eta\exp\left(-\frac{1}{2}[x_t^r - g(x_{t-1}^r,u_t)]^T R^{-1}[x_t^r - g(x_{t-1}^r,u_t)]\right) \tag{41}$$

$$p(z_t^i|x_t) = \eta\exp\left(-\frac{1}{2}[z_t - h(x_t)]^T Q^{-1}[z_t - h(x_t)]\right) \tag{42}$$

$$p(x_0) = \eta\exp\left(-\tfrac{1}{2}[x_0-\mu_0]^T\Sigma_0^{-1}[x_0-\mu_0]\right)\quad \mu_0 = 0,\ \Sigma_0^{-1} = \infty I \tag{43}$$

따라서 해당 값들의 Negative log likelihood는 수식(44)~(46)과 같이, Mahalonobis distance의 형식을 취한다. 여기서 Mahalonobis distance란, 1D 데이터의 경우 해당 데이터가 평균으로 부터 거리가 표준편차의 몇 배인지를 나타내는 값이다. 즉, 해당 값의 신뢰도의 척도이며 이 거리가 크면 클수록 Outlier에 가깝다고 할 수 있다.

$$-\ln p(x_t^r|x_{t-1}^r,u_t) = const. + [x_t^r - g(x_{t-1}^r,u_t)]^T R^{-1}[x_t^r - g(x_{t-1}^r,u_t)] \tag{44}$$

$$-\ln p(z_t^i|x_t) = const. + [z_t - h(x_t)]^T Q^{-1}[z_t - h(x_t)] \tag{45}$$

$$-\ln p(x_0) = const. + [x_0-\mu_0]^T\Sigma_0^{-1}[x_0-\mu_0] \tag{46}$$

따라서 최종적으로 최적화 해야 하는 목적 함수는 식(47)과 같으며, 해당 문제는 제한이 없는 (Unconstrained) 비선형 최적화(Non-linear Optimization) 문제이다.

$$\min_{x_{0:t}} J = \min_{x_{0:t}} \Big[const. + [x_0 - \mu_0]^T \Sigma_0^{-1} [x_0 - \mu_0] + \sum_{\tau=1}^{t} [x_t^r - g(x_{t-1}^r, u_t)]^T R^{-1} [x_t^r - g(x_{t-1}^r, u_t)] + \sum_{\tau=1}^{t} \sum_{i} [z_t - h(x_t)]^T Q^{-1} [z_t - h(x_t)] \Big] \quad (47)$$

3. Iterative Closest Point

그래프 상에서, 연속되는 혹은 연속하지 않는 로봇의 pose로 구성된 node간의 constraint는 odometry 정보와, scan간의 matching으로 만들어 진다. 여기에서, scan matching은 이번 장에서 설명하는 ICP(Iterative closest point)방식으로 진행된다.

3.1. Local Frame to Global Frame

먼저, 로봇의 local frame좌표로 구해진 lidar scan 데이터를 로봇의 현재 pose를 기준으로 global frame에서의 좌표로 변환한다. 이는 식(48)의 roto-translation 변환을 통해 수행된다.

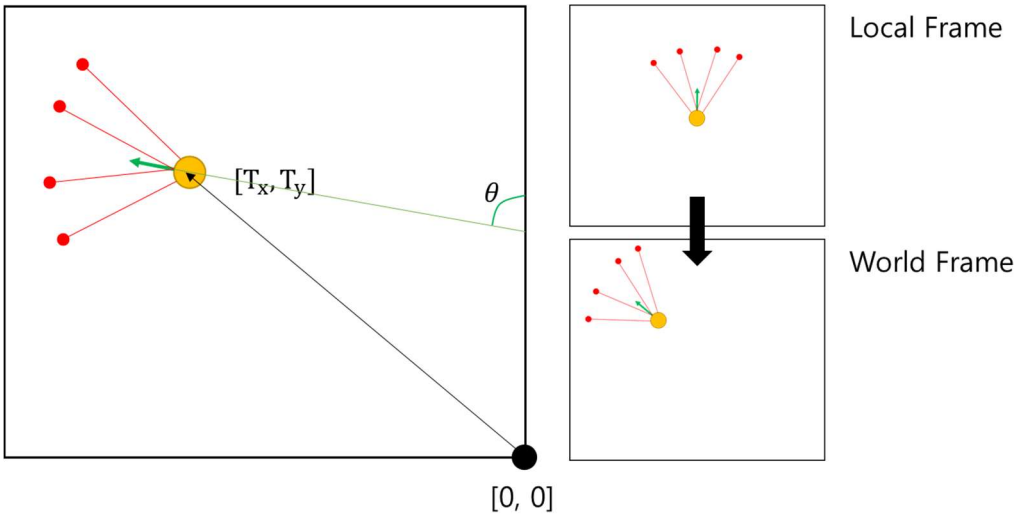

Figure 8 Local Frame과 World Frame

$$\begin{bmatrix} p_{g,x} \\ p_{g,y} \\ 1 \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta & T_x \\ sin\theta & cos\theta & T_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_{l,x} \\ p_{l,y} \\ 1 \end{bmatrix} \quad (48)$$

이렇게 계산된 world frame에서의 lidar scan 데이터의 좌표들은 odometry 센서의 noise로 인해 정확한 위치가 아닐 수 있으며, ICP를 통한 scan matching을 통해 이를 보정한다.

3.2 Scan Matching

Scan matching은 앞서 언급한 odometry 센서의 noise로 인한 차이를 보정하는 방법인데, lidar로부터 새로운 scan 데이터를 획득하여, 이전의 데이터와 가장 잘 일치하는 변환을 찾음으로써 수행된다.

ICP를 통한 scan matching은 연결, 변환, 평가의 세 단계를 거치며, 이는 에러가 수렴할 때 까지 반복된다.

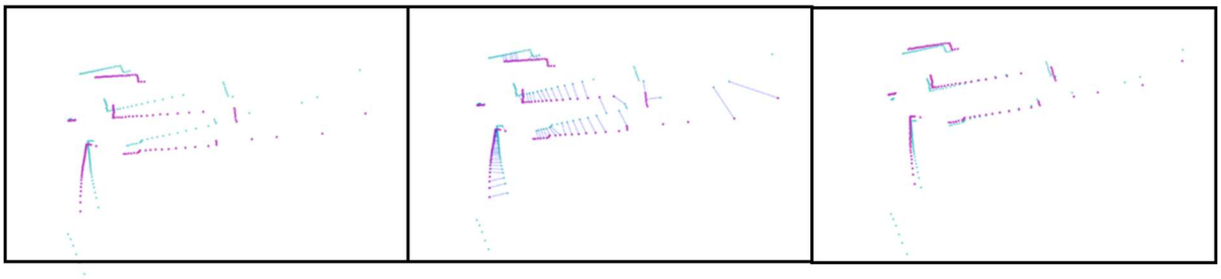

Figure 9 (L)새로운 Scan과 이전 Scan (M)두 Scan간의 Nearest Neighbor (R)Scan Matching

Figure 9의 왼쪽 그림에서 이전 scan 데이터를 하늘색, 새로운 scan 데이터를 보라색이라고 하면, 중간 그림과 같이 두 점들의 집합 간에 가장 거리가 가까운 것들끼리 pair를 만들 수 있다. 이를 통해 식(49)와 같이 이 거리들의 제곱의 합을 최소화 할 수 있는 변환을 찾는다.

$$T^* = \underset{T}{\operatorname{argmin}} \frac{1}{N} \sum_{i}^{N} \left\| \boldsymbol{t}_{i,expand} - T\boldsymbol{s}_{i,expand} \right\|_2^2 \tag{49}$$

식(49)에서 t_i 는 이전 scan 데이터이고 s_i 는 새롭게 측정된 scan 데이터에 해당한다.

3.3 Optimal Rotation

식(49)를 풀기 위해, 먼저 가장 적절한 회전 변환R을 찾는다. 이를 위해 일단 식(50)과 같이 구해진 두 scan 데이터 집합에 각각 해당 scan 데이터의 평균을 빼 주어, 평행이동 시킨다.

$$\boldsymbol{t}_i' = \boldsymbol{t}_i - \boldsymbol{\mu}_t = \begin{bmatrix} t_{i,x} - \mu_{t,x} \\ t_{i,y} - \mu_{t,y} \end{bmatrix}, \boldsymbol{s}_i' = \boldsymbol{s}_i - \boldsymbol{\mu}_s = \begin{bmatrix} s_{i,x} - \mu_{s,x} \\ s_{i,y} - \mu_{s,y} \end{bmatrix} \tag{50}$$

그런 다음 식(51)의 회전에 대한 error를 최소화 하는 회전 변환 R을 구한다.

$$Error(R) = \frac{1}{N} \sum_{i}^{N} \left\| \boldsymbol{t}_i' - R\boldsymbol{s}_i' \right\|_2^2 \tag{51}$$

식(51)을 전개하면 식(52)와 같이 표현할 수 있다.

$$\text{Error(R)} = \frac{1}{N} \sum_{i}^{N} (\left\| \boldsymbol{t}_i' \right\|_2^2 + \left\| R\boldsymbol{s}_i' \right\|_2^2 - 2\boldsymbol{t}_i' R\boldsymbol{s}_i') = -\frac{1}{N} \sum_{i}^{N} 2\boldsymbol{t}_i' R\boldsymbol{s}_i' + const. \tag{52}$$

따라서 이 문제는 식(53)~(54)로 정의할 수 있다.

$$\mathrm{R}^{*} = \underset{R}{\operatorname{argmax}} \sum_{i}^{N} \boldsymbol{t}_{\boldsymbol{i}}' R \boldsymbol{s}_{\boldsymbol{i}}' = \underset{R}{\operatorname{argmax}} \sum_{i}^{N} \operatorname{Trace}\left(\mathrm{R} \boldsymbol{s}_{\boldsymbol{i}}' \boldsymbol{t}_{\boldsymbol{i}}'^{\boldsymbol{T}}\right) \tag{53}$$

$$\text{Let } \mathrm{H} = \boldsymbol{s}_{\boldsymbol{i}}' \boldsymbol{t}_{\boldsymbol{i}}'^{\boldsymbol{T}}$$

$$\mathrm{R}^{*} = \underset{R}{\operatorname{argmax}} \ \operatorname{Trace}(\mathrm{RH}) \tag{54}$$

3.4 Lemma for Optimization

임의의 positive semidefinite matrix AA^{T} 와 임의의 orthonormal matrix B에 대해서 식(55)가 성립한다.

$$\operatorname{Trace}(\mathrm{AA}^{\mathrm{T}}) \geq \ \operatorname{Trace}(\mathrm{BAA}^{\mathrm{T}}) \tag{55}$$

식(55)는 아래와 같이 식(56)~식(59)로 증명할 수 있다.

$$\text{Let } \boldsymbol{a}_{\boldsymbol{i}} \text{ be the } i\text{th column of A}$$

$$\operatorname{Trace}(\mathrm{BAA}^{\mathrm{T}}) = \operatorname{Trace}(\mathrm{A}^{\mathrm{T}}\mathrm{BA}) = \sum_{i} \boldsymbol{a}_{\boldsymbol{i}}^{\boldsymbol{T}} (\mathrm{B} \boldsymbol{a}_{\boldsymbol{i}}) \tag{56}$$

$$\text{by the Schwarz inequallity,} \quad \boldsymbol{a}_{\boldsymbol{i}}^{\boldsymbol{T}} (\mathrm{B} \boldsymbol{a}_{\boldsymbol{i}}) \leq \sqrt{\boldsymbol{a}_{\boldsymbol{i}}^{\boldsymbol{T}} \boldsymbol{a}_{\boldsymbol{i}} (\boldsymbol{a}_{\boldsymbol{i}}^{\boldsymbol{T}} \mathrm{B}^{\mathrm{T}} \mathrm{B} \boldsymbol{a}_{\boldsymbol{i}})} = \boldsymbol{a}_{\boldsymbol{i}}^{\boldsymbol{T}} \boldsymbol{a}_{\boldsymbol{i}} \tag{57}$$

$$\operatorname{Trace}(\mathrm{AA}^{\mathrm{T}}) = \sum_{i} \boldsymbol{a}_{\boldsymbol{i}}^{\boldsymbol{T}} \boldsymbol{a}_{\boldsymbol{i}} \tag{58}$$

$$\because \ \operatorname{Trace}(\mathrm{AA}^{\mathrm{T}}) \geq \operatorname{Trace}(\mathrm{BAA}^{\mathrm{T}}) \tag{59}$$

3.5 Singular Vector Decomposition

식(54)의 H를 SVD(Singular vector decomposition) 하면 식(60)과 같다.

$$\mathrm{H} = \mathrm{U} \Sigma \mathrm{V}^{\mathrm{T}} \tag{60}$$

여기서 singular value는 모두 0 이상이기 때문에, $\mathrm{RH} = \mathrm{VU}^{\mathrm{T}}\mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}} = \mathrm{V}\Sigma\mathrm{V}^{\mathrm{T}}$는 positive semidefinite이고, 따라서 임의의 orthonormal변환 B에 대해서 식 (61)을 만족한다..

$$\operatorname{Trace}(\mathrm{RH}) \geq \operatorname{Trace}(\mathrm{BRH}) \tag{61}$$

orthonormal 변환은 임의의 회전 변환이므로 식(54)를 만족하는 R^{*} 는 VU^{T} 임을 알 수 있다.

따라서 ICP에서는 두 Scan 데이터 간 회전을 VU^{T} 로 계산하며, translation은 간단하게 두 평균간의 거리 $\mathrm{trans} = \mu_t - \mu_s$ 로 계산한다.

Refernece

[1] https://twlab.tistory.com/34

[2] https://twlab.tistory.com/35?category=668741

[3] Steven Waslander, Class Lecture, Topic:"Autonomous Mobile Robot – GraphSLAM." ME597, University of Waterloo.

[4] K. S. Arun, T. S. Huang, and S. D. Blostein. Least square fitting of two 3-d point sets. IEEE Transactions on Pattern Analysis and Machine Intelligence, 9(5):698 – 700, 1987.