Particle Filter Localization

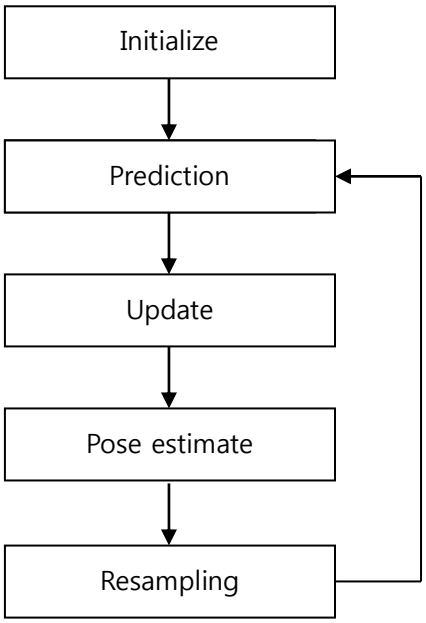
바닥의 패턴을 읽어 Particle Filter로 로봇의 위치를 인식

KITECH 양광웅 작성

호텔 로비나 전시장과 같이 천정이 높고 주위가 넓은 공간에서는 벽이나 천정에 랜드마크를 붙이고 이를 기반으로 로봇의 위치를 파악하는 것은 불가능하다. 이럴 때는 바닥의 패턴을 인식하여 로봇의 위치인식에 활용 가능하다.

Particle Filter는 Bayes filter의 신뢰도, 즉 로봇의 추정 위치를 파티클의 분포로 나타낼 수 있다. 이 방법은 어떤 형태의 확률분포도 나타낼 수 있다. 즉, Kalman filer에서와 같이 로봇의 이동 모델을 근사한 Gaussian으로 가정할 필요가 없으며 선형화 과정도 필요없다.

Particle filter의 수행은 Initialize, Prediction, Update, Normalize, Resampling, Pose Estimation 의 단계로 구성되어 있고, 수행 과정은 아래 그림과 같다.



Particle Filter 알고리즘은 Prediction과 Update 상태를 반복적으로 수행하게 된다. Prediction 상태는 로봇의 이동량에 따라 적절한 에러를 추가하여 파티클을 이동한다. Update 상태는 센서가 측정한 정보에 따라 파티클의 가중치(weight)를 업데이트 한다.

Particle Filter에서 확률분포는 시간 t 에서 가중치를 가지는 N 개의 파티클에 대한 집합 S_t 로 표시된다. $\mathbf{x}_t^{(i)} = [x_t^{(i)}, y_t^{(i)}, \theta_t^{(i)}]^T$ 는 시간 t 에서 i번째 파티클의 상태(위치와 방향)를 나타내고 $w_t^{(i)}$ 는 시간 t 에서 i번째 파티클의 가중치를 의미한다.

$$S_t = \left\{ \left\langle \mathbf{x}_t^{(i)}, w_t^{(i)} \right\rangle | \, i = 1, \cdots, N \right\}$$

Initialize

모든 파티클의 초기값 $\mathbf{x}_{t=0}^{(i)}$ 을 로봇의 알려진 위치 (x, y, θ) 로 동일하게 설정한다. 로봇의 위치를 모를 경우, 지도 전역에 파티클을 랜덤하게 뿌리는 경우도 있지만, 컴퓨터의 계산 성능으로 인한 파티클의 수에 제한이 있기 때문에 파티클이 수렴하지 못하는 경우가 일반적이다. 그래서 파티클 필터로 로봇의 위치를 인식할 때 로봇의 초기 위치를 아는 것이 중요하다.

$$\mathbf{x}_{t=0}^{(i)} \leftarrow (x, y, \theta)$$

$$S_{t=0} = \left\{ \left\langle \mathbf{x}_{t=0}^{(i)}, w_{t=0}^{(i)} \right\rangle | \, i = 1, \cdots, N \right\}$$

여기서 $\sum_{i=1}^{N} w_{t=0}^{(i)} = 1$ 이다.

Prediction

로봇이 u_t 만큼 이동하였을 때, 각 파티클들을 로봇의 이동량 만큼 이동한다. 이때 좌우 바퀴의 회전한 양에 비례하는 노이즈를 더하여 로봇의 이동위치와 방향을 계산한다.

$$\mathbf{x}_t^{(i)} = \mathbf{x}_{t-1}^{(i)} + u_t$$

$$= \begin{bmatrix} x_{t-1}^{(i)} \\ y_{t-1}^{(i)} \\ \theta_{t-1}^{(i)} \end{bmatrix} + \begin{bmatrix} \frac{\Delta s_{r,t}^{(i)} + \Delta s_{l,t}^{(i)}}{2} \cos(\theta_{t-1}^{(i)} + \frac{\Delta s_{r,t}^{(i)} - \Delta s_{l,t}^{(i)}}{2b}) \\ \frac{\Delta s_{r,t}^{(i)} + \Delta s_{l,t}^{(i)}}{2} \sin(\theta_{t-1}^{(i)} + \frac{\Delta s_{r,t}^{(i)} - \Delta s_{l,t}^{(i)}}{2b}) \\ \frac{\Delta s_{r,t}^{(i)} - \Delta s_{l,t}^{(i)}}{b} \end{bmatrix}$$

여기서 b 는 양쪽 바퀴간의 거리다.
노이즈를 고려한 좌우 바퀴의 이동량은 다음과 같다.

$$\Delta s_{r,t}^{(i)} = \text{gauss\_rand}(\Delta s_{r,t},\ k_r \left|\Delta s_{r,t}\right|)$$

$$\Delta s_{l,t}^{(i)} = \text{gauss\_rand}(\Delta s_{l,t},\ k_l \left|\Delta s_{l,t}\right|)$$

여기서 $\Delta s_{r,t}, \Delta s_{l,t}$ 은 마지막 셈플링 기간동안 오른쪽 바퀴와 왼쪽 바퀴가 각각 회전한 양이고, k_r, k_l 은 좌우 바퀴의 에러 상수다. $\text{gauss\_rand}(\mu,\ \Sigma)$ 는 평균 μ 에서 분산 Σ 를 가지는 가우시안 랜덤 값이다.

Update

카메라로 바닥의 패턴을 인식하고 지도와 매칭하여 로봇의 위치를 업데이트 한다.

먼저, 카메라로 인식한 이미지 상의 픽셀 값과 지도 상의 패턴의 밝기 값 간의 유사도를 측정하여 파티클의 확률 $p_t^{(i)}$ 을 계산한다. 유사도 측정 과정은 SSD와 NCC 두 방법 중 하나를 적절하게 골라 사용한다.

SSD(Sum of Squared Deference)

$$SSD = \frac{1}{N}\sqrt{\sum_{j,k}\left(f(x_j,y_k)-t(u_j,v_k)\right)^2}$$

여기서 $f(x_j,y_k)$ 는 x_j,y_k 위치에서 이미지 상의 픽셀 값이고 $t(u_j,v_k)$ 는 u_j,v_k 위치에서 지도 상의 패턴의 밝기 값이다. 그리고 N 은 $j\times k$ 이다. ($0\le f(x_j,y_k)\le 1,\quad 0\le t(u_j,v_k)\le 1$)

$$p_t^{(i)} = 1 - SSD$$

NCC(Normalized Cross Correlation)

$$NCC = \frac{\sum_{j,k}\left(f(x_j,y_k)-\bar{f}\right)\left(t(u_j,v_k)-\bar{t}\right)}{\sqrt{\sum_{j,k}\left(f(x_j,y_k)-\bar{f}\right)^2\sum_{j,k}\left(t(u_j,v_k)-\bar{t}\right)^2}}$$

여기서 $f(x_j,y_k)$ 는 x_j,y_k 위치에서 이미지 상의 픽셀 값이고 $t(u_j,v_k)$ 는 u_j,v_k 위치에서 지도 상의 패턴의 밝기 값이다. 그리고 N 은 $j\times k$ 이다. 그리고 $\bar{f}$ 는 $f(x_j,y_k)$ 의 평균이고 $\bar{t}$ 는 $t(u_j,v_k)$ 의 평균이다. ($-1\le NCC\le 1$)

$$p_t^{(i)} = \frac{NCC+1}{2}$$

매칭 과정이 끝나면, 모든 파티클들의 가중치 합이 1이 되도록 각 파티클의 가중치를 업데이트 한다. 여기서 주의할 점은 현재의 가중치 $w_t^{(i)}$ 는 과거의 가중치 $w_{t-1}^{(i)}$ 에 영향을 받지 않고 매번 새로운 값으로 계산된다는 것이다.

$$w_t^{(i)} = \frac{p_t^{(i)}}{\sum_{k=1}^{N} p_t^{(k)}}$$

이제 기존 파티클의 집합 S 에 대한 Prediction과 Update 과정이 끝나 새로운 상태가 되었다.

$$S_t = \left\{\left\langle \mathbf{x}_t^{(i)}, w_t^{(i)}\right\rangle : i = 1,\cdots,N\right\}$$

Pose Estimate

모든 파티클들의 가중치와 위치의 곱을 합하여 최종 로봇의 추정 위치 $\mathbf{p}_{est}$ 를 계산한다.

$$\mathbf{p}_{est} = \sum_{i=1}^{N} w_t^{(i)} \mathbf{x}_t^{(i)}$$

이때 θ 의 연산에 주의하여야 한다. 예를 들자면, 각도 -180과 180을 더하여 평균을 내면 0이지만, 실제로 평균은 180이 되어야 한다. θ 는 -180도 에서 +180도의 값을 가지므로 +영역에 있는 각도와 −영역에 있는 각도를 따로 계산하는데, −영역의 값을 기준으로 +영역의 값과 −영역의 값의 차를 더하여 평균을 계산한다.

Resampling

각 파티클들의 가중치에 비례하는 새로운 파티클들을 생성한다. 즉, 가중치가 작은 파티클은 소멸하게 되고 가중치가 클수록 많은 새로운 파티클들을 생성한다. 새로운 파티클은 기존 파티클의 특성 $(\mathbf{x}_t^{(i)}, w_t^{(i)})$ 을 그대로 물려받는다.

파티클 i 가 만들어내는 새로운 파티클의 수 n 은 다음과 같이 계산한다.

$$n = \left\lceil N w_t^{(i)} \right\rceil$$

[첨부] Map Example