Introducción a Robotic Operating System

Estimación y SLAM

Martín Llofriu y Gonzalo Tejera
Facultad de Ingeniería :: Instituto de Computación

mina
network management | artificial intelligence

Contenido

- El problema de localización
- Filtro de partículas
- Integrando odometría
- Integrando medidas

El problema de localización

Empecemos por un problema hermano: Tracking

El problema de tracking consiste en estimar la posición de un objeto en movimiento a partir de:

- Un modelo de como evoluciona la posición con el tiempo
- Observaciones

Tracking: Modelo

El modelo de un objeto moviendose a velocidad constante sería:

dx = v * cos(theta)

dy = v * sin(theta)

theta = theta

El modelo puede:

- Ser impreciso: es difícil modelar todos los factores que inciden en un proceso cinemático
- Contener una componente de ruido que no conocemos: el auto puede no avanzar exactamente en proporción a la aceleración comandada

Tracking: Observaciones

- Las observaciones pueden ser ruidosas
- Las observaciones pueden no ser tan frecuentes como la actualización de odometría

Tracking: Integración

Cómo integrar toda esta información?

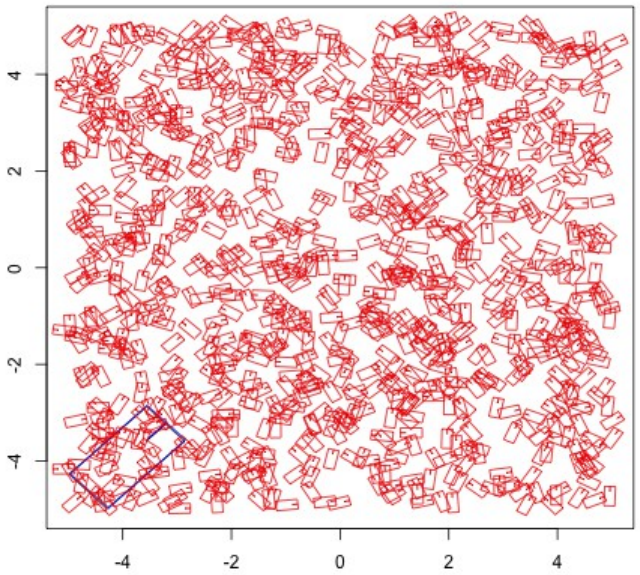
Si confiamos sólo en las medidas, nuestra estimación es ruidosa, sensible a *outliers* y puede tener demoras

Si confiamos solo en el modelo, nuestra estimación acumula error a través del tiempo

GIFs

Partículas

- Estimador que consta de un conjunto de *samples*
- Cada uno de estos *samples* consta de una posicion (x,y,theta) del robot



Init

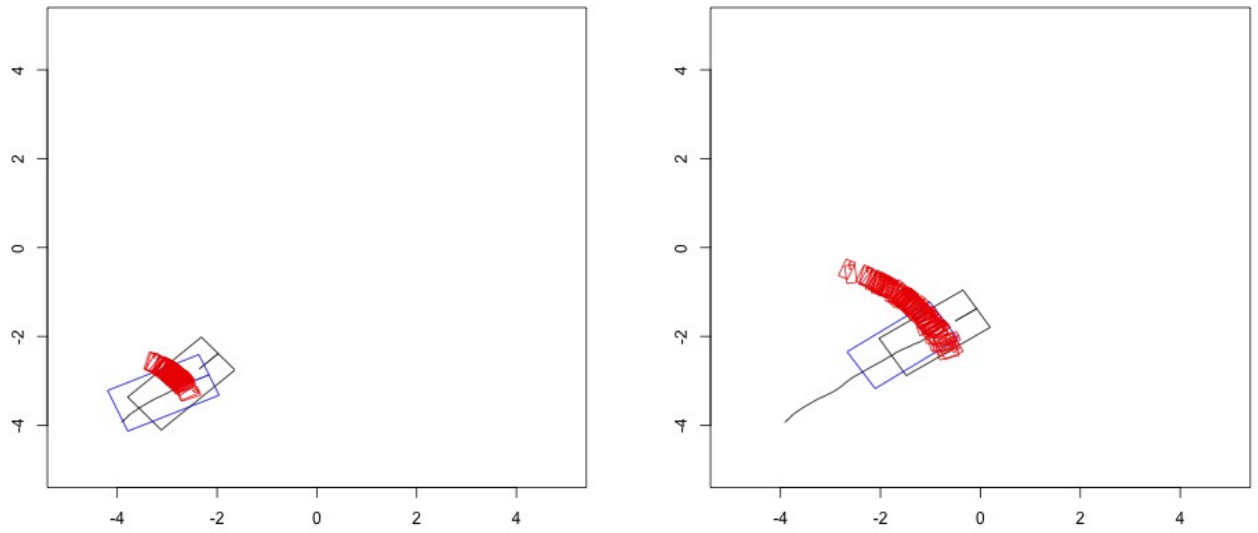
- Cada partícula es inicializada con una posición al azar o en el entorno de una posición inicial conocida

Init

```
for p in particles:
        p.x = init_x
        p.y = init_y
        p.theta = init_theta
```

Predict

- Luego de cada actualización de la información de odometría cada partícula se mueve de acuerdo al cambio de posición



Predict

Predict

```
# input: particles, odom, motionModel

for p in particles:
        p.x = p.x + odom.x + motionModel.sampleNoiseX()
        p.y = p.y + odom.y + motionModel.sampleNoiseY()
        p.theta = p.theta + odom.theta + motionModel.sampleNoiseTheta()

# output: updated particles
```

Update

- Cada vez que se recibe información de una observación, se depuran las partículas
- Cada partícula es asignada con un peso en proporción a su coherencia con la observación realizada

Update

Update

```
# input: particles, obs, observationModel

weights = list(0)
for p in particles:
        weights[p] = observationModel.calcProb(p, obs)

particles = resample(particles, weights)

# output: resampled particles
```

Update - Resample

- Un nuevo conjunto de partículas es elegido, eligiendo del conjunto original con reposición
- La probabilidad de elegir una partícula es proporcional a su peso

Update - Resample

Resample

```
# input: particles, weights

weights = normalize (weights)

newParticles = list()

for (i in 1:numParticles):
        r = random()
        j = 0
        while (r - weights(j) > 0):
                j++
                r = r - weights(j)
        newParticles[i] = particles[j]

particles = newParticles

# output: resampled particles
```

GIFs

SLAM

- Simultaneous Localization and Mapping
- Armar un mapa a la vez que nos ubicamos en el
- Cada partícula va a estimar la posición del robot y de cada marca al mismo tiempo
- Cada partícula tendrá asociados la posición del robot y de las marcas vistas

Partículas en ROS

- Podemos usar transformadas para mantener la posición de cada partícula
- Usamos una convención de nombres: “/p1” es la transformada de la partícula p1
- Predict: actualizamos las transformadas /p1-100

Partículas en ROS

- Podemos representar las marcas como transformadas: “/pi/mj” para la marca j de la partícula i
- Update:
  - Transformamos la observación al marco absoluto, p.e. map
  - Medimos la diferencia entre transformadas
  - Asignamos peso en proporción a esta diferencia
- Para medir la diferencia entre transformadas:
  - “Multiplicamos” una por la inversa de otra
  - Medimos la magnitud de la transformación obtenida

Video

https://www.youtube.com/watch?v=omGNeV3VYUY