Gmapping

算法原理

RBPF概述: 一种SLAM方法, 将定位和建图分离, 通过带有权重、轨迹、地图的散布的粒子来表示地图和定位的后验概率分布. (*类似采样枚举法, 只采样枚举可能性高的点*)
Gmapping概述: 基于RBPF, 通过加入观测优化了提议分布, 通过选择性重采样, 降低了采样频率.

关键词: 目标分布、提议分布、选择性重采样、粒子退化.

详解

贝叶斯滤波

基本贝叶斯估计知识:

独立:

$$p(x,y) = p(x)p(y)$$
$$p(x|y) = p(x)$$
$$p(y|x) = p(y)$$

条件独立:

$$p(x,y|z) = p(x|z)p(y|z)$$
$$p(x|y,z) = p(x|z)$$
$$p(y|x,z) = p(y|z)$$

全概率公式:

$$p(x) = \int p(x|y)p(y)dy$$

条件概率公式:

$$P(x,y) = P(x|y)P(y) = P(y|x)P(x)$$

贝叶斯公式:

$$p(x,y) = p(x|y)p(y) = p(y|x)p(x)$$

$$\rightarrow p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \eta p(y|x)p(x)$$

条件贝叶斯公式:

$$p(x|y,z) = \frac{p(y|x,z)p(x|z)}{p(y|z)} = \eta p(y|x,z)p(x|z)$$

贝叶斯滤波: 已知状态量$t-1$时刻的概率分布, 在给定t时刻的观测数据z_t, u_t的情况下, 估计出状态量在t时刻的概率分布.

$$bel(x_t) = p(x_t|z_{1:t}, u_{1:t}) = \eta p(z_t|x_t)\overline{bel}(x_t) = \eta p(z_t|x_t)\int p(x_t|x_{t-1}, u_t)p(x_{t-1}|z_{1:t-1}, u_{1:t-1})dx_{t-1}$$

Rao-Blackwellized Particle Filters

定位和建图分离:

$$p(x_{1:t}, m|z_{1:t}, u_{1:t-1}) = p(m|x_{1:t}, z_{1:t}) \cdot p(x_{1:t}|z_{1:t}, u_{1:t-1})$$

在粒子滤波器中, 后验概率分布的样本称为粒子particles

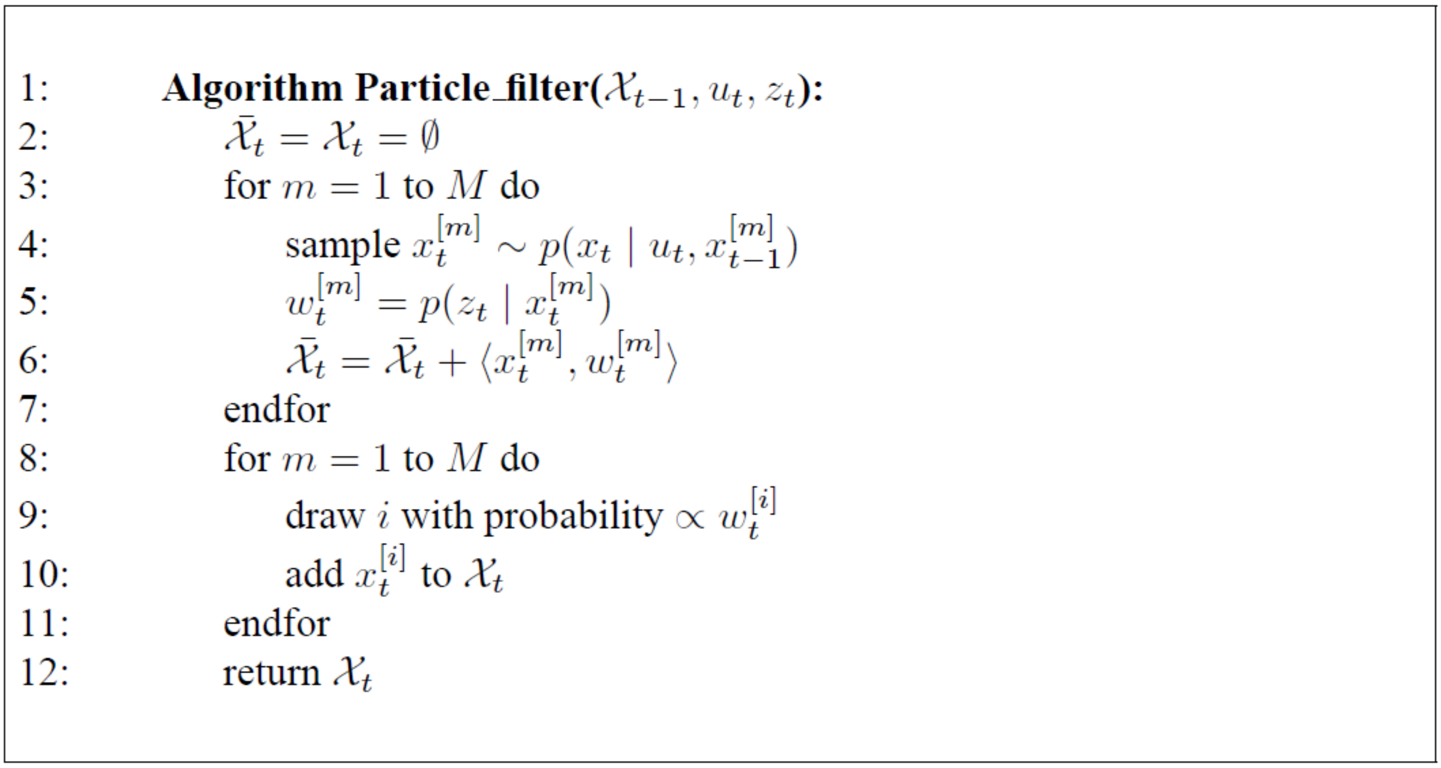
$$\mathcal{X}_t := x_t^{[1]}, x_t^{[2]}, \cdots, x_t^{[M]}$$

其中$\mathcal{X}_t$为粒子集合, 每一个粒子$x_t^{[m]}, 1 \leq m \leq M$, 都是状态在t时刻的一个具体实例, 也就是在t时刻真实世界可能状态的一个假设.

粒子滤波器背后的思想是用集合$\mathcal{X}_t$来近似置信度$bel(x_t)$, 理想情况下粒子集合$\mathcal{X}_t$中含有状态假设x_t的似然概率应当与贝叶斯滤波器的后验概率$bel(x_t)$成正比关系:

$$x_t^{[m]} \sim p(x_t|z_{1:t}, u_{1:t})$$

算法伪代码:



```
1:  Algorithm Particle_filter($\mathcal{X}_{t-1}, u_t, z_t$):
2:      $\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
3:      for $m = 1$ to $M$ do
4:          sample $x_t^{[m]} \sim p(x_t \mid u_t, x_{t-1}^{[m]})$
5:          $w_t^{[m]} = p(z_t \mid x_t^{[m]})$
6:          $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
7:      endfor
8:      for $m = 1$ to $M$ do
9:          draw $i$ with probability $\propto w_t^{[i]}$
10:         add $x_t^{[i]}$ to $\mathcal{X}_t$
11:     endfor
12:     return $\mathcal{X}_t$
```

RBPF存在两大问题: 粒子数多(造成计算量和内存消耗变大)、频繁重采样(加剧了粒子退化).
粒子退化: 主要指正确的粒子被丢弃和粒子多样性减小, 而频繁重采样则加剧了正确的粒子被丢弃的可能性和粒子多样性减小速率.
粒子权重的计算:

$$\begin{aligned}
w_t^{(i)} &= \frac{p(x_{1:t}^{(i)}|z_{1:t}, u_{1:t-1})}{\pi(x_{1:t}^{(i)}|z_{1:t}, u_{1:t-1})} \\
&= \frac{\eta p(z_t|x_{1:t}^{(i)}, z_{1:t-1})p(x_t^{(i)}|x_{t-1}^{(i)}, u_{t-1})}{\pi(x_t^{(i)}|x_{1:t-1}^{(i)}, z_{1:t}, u_{1:t-1})} \cdot \underbrace{\frac{p(x_{1:t-1}^{(i)}|z_{1:t-1}, u_{1:t-2})}{\pi(x_{1:t-1}^{(i)}|z_{1:t-1}, u_{1:t-2})}}_{w_{t-1}^{(i)}} \\
&\propto \frac{p(z_t|m_{t-1}^{(i)}, x_t^{(i)})p(x_t^{(i)}|x_{t-1}^{(i)}, u_{t-1})}{\pi(x_t|x_{1:t-1}^{(i)}, z_{1:t}, u_{1:t-1})} \cdot w_{t-1}^{(i)}
\end{aligned}$$

$\eta = 1/p(z_t|z_{1:t-1}, u_{1:t-1})$ 是一个归一化参数, 每个粒子都一样.
p是目标分布: 根据机器人携带的所有传感器的数据能确定机器人状态置信度的最大极限.
π是提议分布: 由于无法直接对目标分布建模进行采样, 比如对于激光传感器, 无法进行高斯建模. 但问题是我们希望从一个分布中进行采样来获取对下一时刻机器人位姿的估计, 而在计算机中能模拟出的分布也就是高斯分布、三角分布等有限的分布, 因此提议分布被提出来代替目标分布来提取下一时刻机器人位姿信息. 而提议分布毕竟不是目标分布因此使用粒子权重来表征提议分布和目标分布的不一致性. [refer to GMapping原理分析]

Gmapping

Fast Slam以motion model作为提议分布:

$$\begin{aligned}
w_t^{(i)} &= w_{t-1}^{(i)} \frac{\eta p(z_t|m_{t-1}^{(i)}, x_t^{(i)})p(x_t^{(i)}|x_{t-1}^{(i)}, u_{t-1})}{p(x_t^{(i)}|x_{t-1}^{(i)}, u_{t-1})} \\
&\propto w_{t-1}^{(i)} \cdot p(z_t|m_{t-1}^{(i)}, x_t^{(i)})
\end{aligned}$$

改进, 以激光scanmatch作为提议分布:

$$p(x_t|m_{t-1}^{(i)}, x_{t-1}^{(i)}, z_t, u_{t-1}) = \frac{p(z_t|m_{t-1}^{(i)}, x_t)p(x_t|x_{t-1}^{(i)}, u_{t-1})}{p(z_t|m_{t-1}^{(i)}, x_{t-1}^{(i)}, u_{t-1})}$$

注: 为什么要除这一项? 我们要的是x_t的分布, 而分子上是$z_t x_t$的分布.
从而有:

$$
\begin{aligned}
w_t^{(i)} &= w_{t-1}^{(i)} \frac{\eta p(z_t|m_{t-1}^{(i)}, x_t^{(i)}) p(x_t^{(i)}|x_{t-1}^{(i)}, u_{t-1})}{p(x_t|m_{t-1}^{(i)}, x_{t-1}^{(i)}, z_t, u_{t-1})} \\
&\propto \quad w_{t-1}^{(i)} \frac{p(z_t|m_{t-1}^{(i)}, x_t^{(i)}) p(x_t^{(i)}|x_{t-1}^{(i)}, u_{t-1})}{\frac{p(z_t|m_{t-1}^{(i)}, x_t) p(x_t|x_{t-1}^{(i)}, u_{t-1})}{p(z_t|m_{t-1}^{(i)}, x_{t-1}^{(i)}, u_{t-1})}} \\
&= w_{t-1}^{(i)} \cdot p(z_t|m_{t-1}^{(i)}, x_{t-1}^{(i)}, u_{t-1}) \\
&= w_{t-1}^{(i)} \cdot \int p(z_t|x') p(x'|x_{t-1}^{(i)}, u_{t-1}) dx' \\
&\simeq w_{t-1}^{(i)} \cdot \sum_{j=1}^{K} p(z_t|m_{t-1}^{(i)}, x_j) \cdot p(x_j|x_{t-1}^{(i)}, u_{t-1})
\end{aligned}
$$

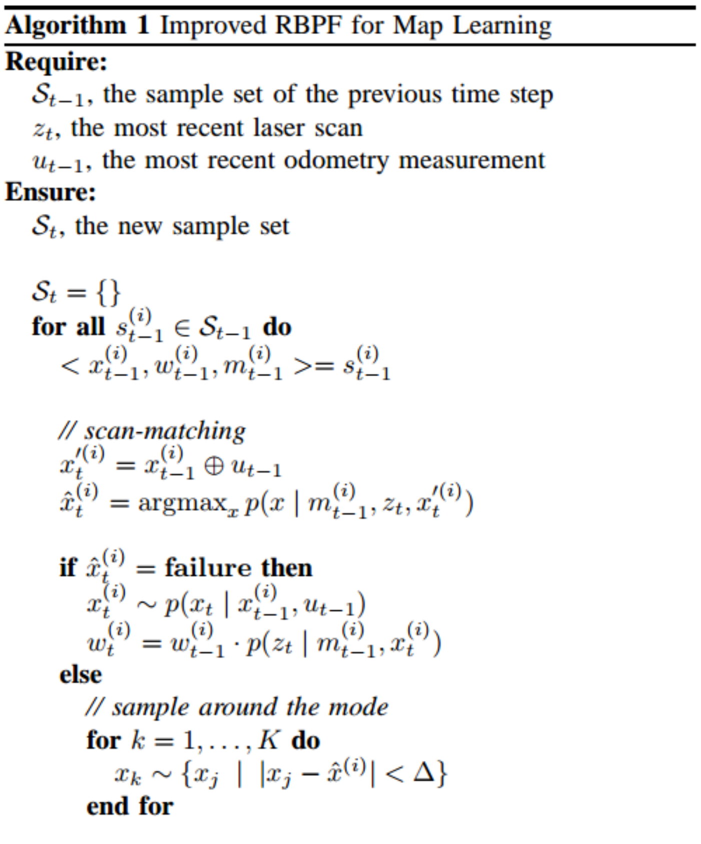
在scanmatch的有效区域采样K个点, 计算μ和Σ, 得到提议高斯分布:

$$
\mu_t^{(i)} = \frac{1}{\eta^{(i)}} \cdot \sum_{j=1}^{K} x_j \cdot p(z_t|m_{t-1}^{(i)}, x_j) \cdot p(x_j|x_{t-1}^{(i)}, u_{t-1})
$$

$$
\Sigma_t^{(i)} = \frac{1}{\eta^{(i)}} \cdot \sum_{j=1}^{K} p(z_t|m_{t-1}^{(i)}, x_j) \cdot p(x_j|x_{t-1}^{(i)}, u_{t-1}) (x_j - \mu_t^{(i)})(x_j - \mu_t^{(i)})^T
$$

若后一刻的观测和前一刻的结论一致性很好, 则粒子权重变化很小, 但若后一刻的观测和前一刻的结论出现较大偏差(比如闭环时候), 粒子的权重会发生剧变. 体现在(或许scanmatch配上情况比例发生变化).

算法伪代码:



Algorithm 1 Improved RBPF for Map Learning

```
Require:
  $\mathcal{S}_{t-1}$, the sample set of the previous time step
  $z_t$, the most recent laser scan
  $u_{t-1}$, the most recent odometry measurement
Ensure:
  $\mathcal{S}_t$, the new sample set

  $\mathcal{S}_t = \{\}$
  for all $s_{t-1}^{(i)} \in \mathcal{S}_{t-1}$ do
    $< x_{t-1}^{(i)}, w_{t-1}^{(i)}, m_{t-1}^{(i)} >= s_{t-1}^{(i)}$

    // scan-matching
    $x_t'^{(i)} = x_{t-1}^{(i)} \oplus u_{t-1}$
    $\hat{x}_t^{(i)} = \text{argmax}_x\, p(x \mid m_{t-1}^{(i)}, z_t, x_t'^{(i)})$

    if $\hat{x}_t^{(i)}$ = failure then
      $x_t^{(i)} \sim p(x_t \mid x_{t-1}^{(i)}, u_{t-1})$
      $w_t^{(i)} = w_{t-1}^{(i)} \cdot p(z_t \mid m_{t-1}^{(i)}, x_t^{(i)})$
    else
      // sample around the mode
      for $k = 1, \ldots, K$ do
        $x_k \sim \{x_j \mid |x_j - \hat{x}^{(i)}| < \Delta\}$
      end for
      // compute Gaussian proposal
      $\mu_t^{(i)} = (0,0,0)^T$
      $\eta^{(i)} = 0$
      for all $x_j \in \{x_1, \ldots, x_K\}$ do
        $\mu_t^{(i)} = \mu_t^{(i)} + x_j \cdot p(z_t \mid m_{t-1}^{(i)}, x_j) \cdot p(x_t \mid x_{t-1}^{(i)}, u_{t-1})$
        $\eta^{(i)} = \eta^{(i)} + p(z_t \mid m_{t-1}^{(i)}, x_j) \cdot p(x_t \mid x_{t-1}^{(i)}, u_{t-1})$
      end for
      $\mu_t^{(i)} = \mu_t^{(i)} / \eta^{(i)}$
      $\Sigma_t^{(i)} = \mathbf{0}$
      for all $x_j \in \{x_1, \ldots, x_K\}$ do
        $\Sigma_t^{(i)} = \Sigma_t^{(i)} + (x_j - \mu^{(i)})(x_j - \mu^{(i)})^T \cdot$
                  $p(z_t \mid m_{t-1}^{(i)}, x_j) \cdot p(x_j \mid x_{t-1}^{(i)}, u_{t-1})$
      end for
      $\Sigma_t^{(i)} = \Sigma_t^{(i)} / \eta^{(i)}$
      // sample new pose
      $x_t^{(i)} \sim \mathcal{N}(\mu_t^{(i)}, \Sigma_t^{(i)})$

      // update importance weights
      $w_t^{(i)} = w_{t-1}^{(i)} \cdot \eta^{(i)}$
    end if
    // update map
    $m_t^{(i)} = \mathrm{integrateScan}(m_{t-1}^{(i)}, x_t^{(i)}, z_t)$
    // update sample set
    $\mathcal{S}_t = \mathcal{S}_t \cup \{< x_t^{(i)}, w_t^{(i)}, m_t^{(i)} >\}$
  end for

  $N_{\text{eff}} = \frac{1}{\sum_{i=1}^{N} (\tilde{w}^{(i)})^2}$

  if $N_{\text{eff}} < T$ then
    $\mathcal{S}_t = \mathrm{resample}(\mathcal{S}_t)$
  end if
```

关键代码

关键代码位于 `openslam_gmapping/gridfastslam/gridslamprocessor.cpp` 之中:

```cpp
bool GridSlamProcessor::processScan(const RangeReading &reading, int adaptParticles) {
        //write the state of the reading and update all the particles using the motion model
        for (ParticleVector::iterator it = m_particles.begin(); it != m_particles.end(); it++) {
            OrientedPoint &pose(it->pose);
            pose = m_motionModel.drawFromMotion(it->pose, relPose, m_odoPose);
        }

        /*
         * 为每个粒子进行scanMatch，计算出来每个粒子的最优位姿，同时计算改最优位姿的得分和似然
         * 对应于gmapping论文中的用最近的一次测量计算proposal的算法
         * 这里面除了进行scanMatch之外，还对粒子进行了权重的计算，
         * 并计算了粒子的有效区域 但不进行内存分配 内存分配在resample()函数中
         * 这个函数在gridslamprocessor.hxx里面
         */
        scanMatch(plainReading);

        // 由于scanMatch中对粒子的权重进行了更新，那么这个时候各个粒子的轨迹上的累计权重都需要重新计算
        // 这个函数即更新各个粒子的轨迹上的累计权重是更新
        updateTreeWeights(false);

        // 粒子重采样  根据neff的大小来进行重采样  不但进行了重采样，也对地图进行更新
        // GridSlamProcessor::resample 函数在gridslamprocessor.hxx里面实现
        resample(plainReading, adaptParticles, reading_copy);
}
```

安装运行

ROS中的slam_gmapping包也是调用了openslam_gmapping开源算法, 因此需要下载编译两个代码.

```
mkdir gmapping_ws
cd gmapping_ws
mkdir src
cd src
git clone https://github.com/ros-perception/slam_gmapping
git clone https://github.com/ros-perception/openslam_gmapping
cd ..
catkin_make
```

订阅消息:

```
scan : 2d激光数据，注意正反角度顺序
tf : 里程数据
```

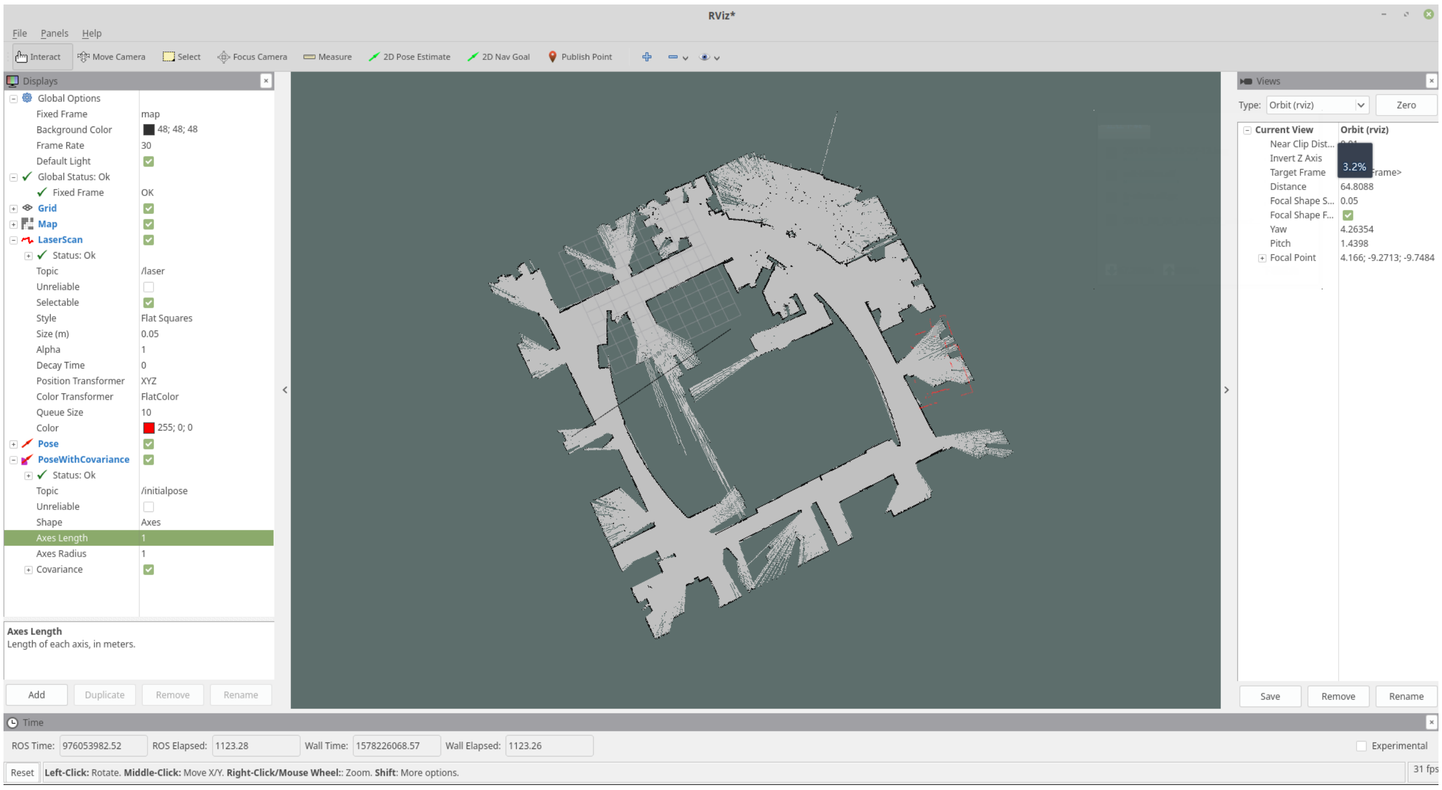
将数据转化为gmapping的rosbag

```
cd
```

```
cd /media/wegatron/data/workspace/opensource_poj/gmapping_ws
roslaunch gmapping basic_localization_stage.launch
```

测试分析

intel lab:



Global Options	
Fixed Frame	map
Background Color	⬛ 48; 48; 48
Frame Rate	30
Default Light	✓
Global Status: Ok	
Fixed Frame	OK
Grid	✓
Map	✓
LaserScan	✓
Status: Ok	
Topic	/laser
Unreliable	☐
Selectable	✓
Style	Flat Squares
Size (m)	0.05
Alpha	1
Decay Time	0
Position Transformer	XYZ
Color Transformer	FlatColor
Queue Size	10
Color	🟥 255; 0; 0
Pose	✓
PoseWithCovariance	✓
Status: Ok	
Topic	/initialpose
Unreliable	☐
Shape	Axes
Axes Length	1
Axes Radius	1
Covariance	✓

Axes Length
Length of each axis, in meters.

Type: Orbit (rviz)

Current View	Orbit (rviz)
Near Clip Dist...	[illegible]
Invert Z Axis	
Target Frame	[illegible]Frame>
Distance	64.8088
Focal Shape S...	0.05
Focal Shape F...	✓
Yaw	4.26354
Pitch	1.4398
Focal Point	4.166; -9.2713; -9.7484

ROS Time: 976053982.52 ROS Elapsed: 1123.28 Wall Time: 1578226068.57 Wall Elapsed: 1123.26

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift:** More options.

Reference

数据集运行Gmapping
SLAM构建地图——gmapping功能包
GMapping原理分析
粒子滤波原理分析
gmapping源码阅读
intel lab data to ros bag for gmapping
mit 数据集