ros的navigation之—Amcl（localization）应用详解

发布时间：2016年10月27日 22:21:49 浏览数：1793次 来自：IT大道 (http://www.itdadao.com/articles/c15a641272p0.html)

amcl的英文全称是adaptive Monte Carlo localization，其实就是蒙特卡洛定位方法的一种升级版，使用自适应的KLD方法来更新粒子，这里不再多说（主要我也不熟
趣的可以去看：KLD。 ...

关于amcl

amcl的英文全称是adaptive Monte Carlo localization，其实就是蒙特卡洛定位方法的一种升级版，使用自适应的KLD方法来更新粒子，这里不再多
我也不熟），有兴趣的可以去看：KLD (https://en.wikipedia.org/wiki/Kullback%E2%80%93Leibler_divergence)。
而amcl（蒙特卡洛定位）法使用的是粒子滤波的方法来进行定位的。而粒子滤波很粗浅的说就是一开始在地图空间很均匀的撒一把粒子，然后通过
的motion来移动粒子，比如机器人向前移动了一米，所有的粒子也就向前移动一米，不管现在这个粒子的位置对不对。使用每个粒子所处位置模拟
信息跟观察到的传感器信息（一般是激光）作对比，从而赋给每个粒子一个概率。之后根据生成的概率来重新生成粒子，概率越高的生成的概率越大
代之后，所有的粒子会慢慢地收敛到一起，机器人的确切位置也就被推算出来了。

(data/attachment/portal/201610/27/221346vpbss08tb3co0f5t.jpg)

这幅图模拟了一个一维机器人的粒子更新，机器人下面那些想条形码一样的竖条就是粒子的分布了，可以看到粒子随着机器人的移动与更新会逐渐的
人的正确位置上。
amcl算法步骤图：

Algorithm MCL(X_{t-1}, u_t, z_t):
$\bar{X}_t = X_t = \emptyset$
for $m = 1$ to M:
$x_t^{[m]} = \textbf{motion\_update}(u_t, x_{t-1}^{[m]})$
$w_t^{[m]} = \textbf{sensor\_update}(z_t, x_t^{[m]})$
$\bar{X}_t = \bar{X}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
endfor
for $m = 1$ to M:
draw $x_t^{[m]}$ from $\bar{X}_t$ with probability $\propto w_t^{[m]}$
$X_t = X_t + x_t^{[m]}$
endfor
return X_t

wiki链接 (https://en.wikipedia.org/wiki/Monte_Carlo_localization)

ros中的amcl

ros中使用的就是自适应的蒙特卡洛定位法。

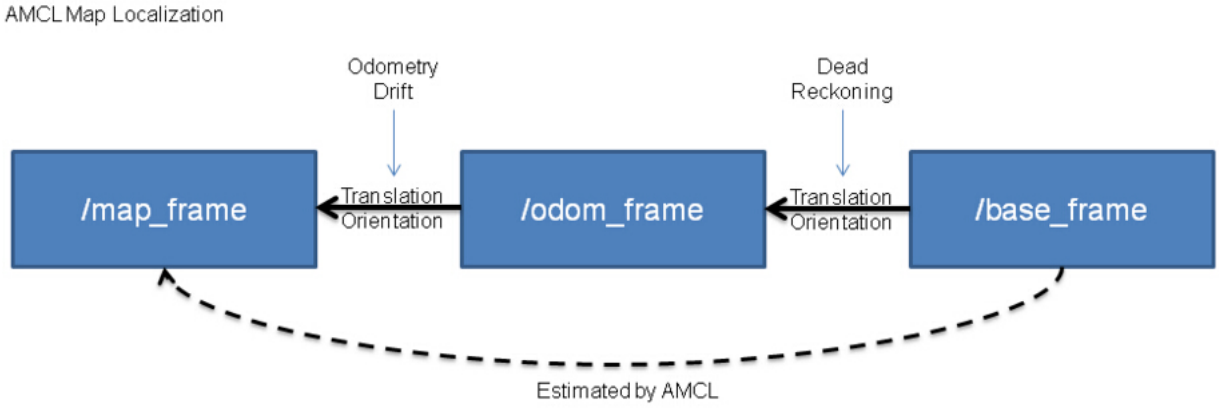
> 订阅的主题

scan (sensor_msgs/LaserScan)
Laser scans.
tf (tf/tfMessage)
Transforms.
initialpose (geometry_msgs/PoseWithCovarianceStamped)
Mean and covariance with which to (re-)initialize the particle filter.
map (nav_msgs/OccupancyGrid)
When the use_map_topic parameter is set, AMCL subscribes to this topic to retrieve the map used for laser-based localization. New in nav
1.4.2.
实际上初始位姿可以通过参数提供也可以使用默认初始值，我们主要是要将scan（激光）、tf和map主题提供给amcl。

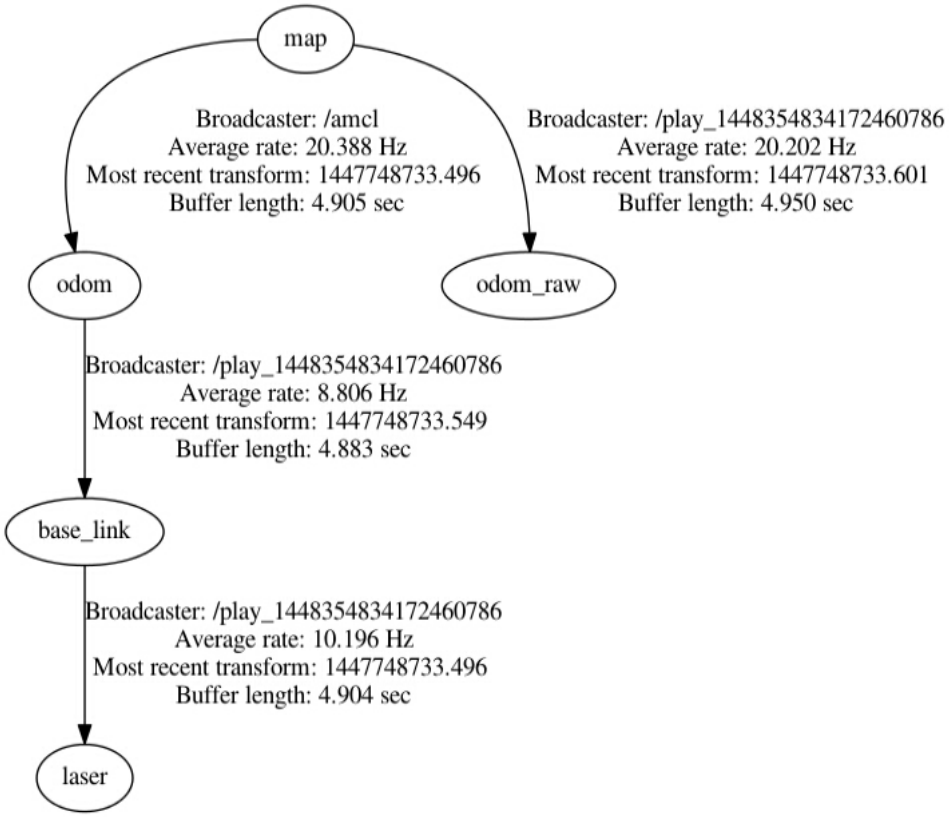
> 发布的主题

amcl_pose (geometry_msgs/PoseWithCovarianceStamped)
Robot’s estimated pose in the map, with covariance.
particlecloud (geometry_msgs/PoseArray)
The set of pose estimates being maintained by the filter.
tf (tf/tfMessage)
Publishes the transform from odom (which can be remapped via the ~odom_frame_id parameter) to map.
如果纯粹是为了显示一下机器人的位姿（in rviz）我们只需要tf主题就够了。

> tf tree



我们需要将base_link，odom，map三个frame连接起来。

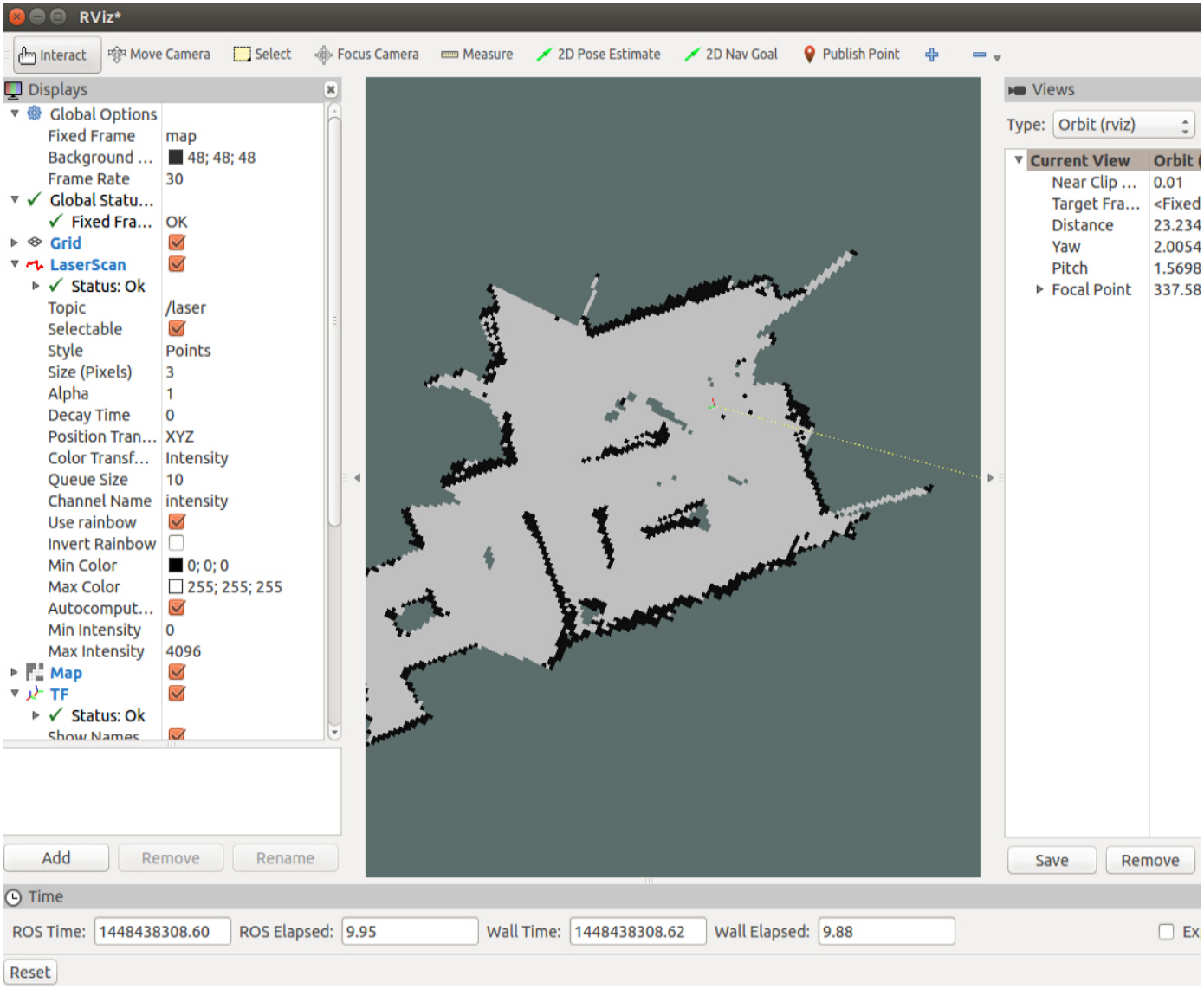


上图是我自己的tf连接图

> example

```
<?xml version="1.0"?>
<launch>
   <!-->
  <node pkg="beginner_tutorials" type="talker" name="talker"/>
   <-->
  <node pkg="map_server" type="map_server" name="map_server" args="/home/zqq/map.yaml"/>

  <!-- amcl node -->
  <node pkg="amcl" type="amcl" name="amcl" output="screen">

  <remap from="scan" to="scan"/>
  <!-- Publish scans from best pose at a max of 10 Hz -->
  <param name="use_map_topic" value="http://blog.csdn.net/chenxingwangzi/article/details/true"/>
  <param name="odom_model_type" value="http://blog.csdn.net/chenxingwangzi/article/details/omni"/>
  <param name="odom_alpha5" value="http://blog.csdn.net/chenxingwangzi/article/details/0.1"/>
  <param name="transform_tolerance" value="http://blog.csdn.net/chenxingwangzi/article/details/0.5" />
  <param name="gui_publish_rate" value="http://blog.csdn.net/chenxingwangzi/article/details/10.0"/>
  <param name="laser_max_beams" value="http://blog.csdn.net/chenxingwangzi/article/details/300"/>
  <param name="min_particles" value="http://blog.csdn.net/chenxingwangzi/article/details/500"/>
  <param name="max_particles" value="http://blog.csdn.net/chenxingwangzi/article/details/5000"/>
  <param name="kld_err" value="http://blog.csdn.net/chenxingwangzi/article/details/0.1"/>
  <param name="kld_z" value="http://blog.csdn.net/chenxingwangzi/article/details/0.99"/>
  <param name="odom_alpha1" value="http://blog.csdn.net/chenxingwangzi/article/details/0.1"/>
  <param name="odom_alpha2" value="http://blog.csdn.net/chenxingwangzi/article/details/0.1"/>
  <!-- translation std dev, m -->
  <param name="odom_alpha3" value="http://blog.csdn.net/chenxingwangzi/article/details/0.1"/>
  <param name="odom_alpha4" value="http://blog.csdn.net/chenxingwangzi/article/details/0.1"/>
  <param name="laser_z_hit" value="http://blog.csdn.net/chenxingwangzi/article/details/0.9"/>
  <param name="laser_z_short" value="http://blog.csdn.net/chenxingwangzi/article/details/0.05"/>
  <param name="laser_z_max" value="http://blog.csdn.net/chenxingwangzi/article/details/0.05"/>
  <param name="laser_z_rand" value="http://blog.csdn.net/chenxingwangzi/article/details/0.5"/>
  <param name="laser_sigma_hit" value="http://blog.csdn.net/chenxingwangzi/article/details/0.2"/>
  <param name="laser_lambda_short" value="http://blog.csdn.net/chenxingwangzi/article/details/0.1"/>
  <param name="laser_lambda_short" value="http://blog.csdn.net/chenxingwangzi/article/details/0.1"/>
  <param name="laser_model_type" value="http://blog.csdn.net/chenxingwangzi/article/details/likelihood_field"/>
  <!-- <param name="laser_model_type" value="http://blog.csdn.net/chenxingwangzi/article/details/beam"/> -->
 <param name="laser_min_range" value="http://blog.csdn.net/chenxingwangzi/article/details/1"/>
<param name="laser_max_range" value="http://blog.csdn.net/chenxingwangzi/article/details/5"/>
  <param name="laser_likelihood_max_dist" value="http://blog.csdn.net/chenxingwangzi/article/details/2.0"/>
  <param name="update_min_d" value="http://blog.csdn.net/chenxingwangzi/article/details/0.2"/>
  <param name="update_min_a" value="http://blog.csdn.net/chenxingwangzi/article/details/0.5"/>
  <param name="resample_interval" value="http://blog.csdn.net/chenxingwangzi/article/details/1"/>
  <param name="transform_tolerance" value="http://blog.csdn.net/chenxingwangzi/article/details/0.1"/>
  <param name="recovery_alpha_slow" value="http://blog.csdn.net/chenxingwangzi/article/details/0.0"/>
  <param name="recovery_alpha_fast" value="http://blog.csdn.net/chenxingwangzi/article/details/0.0"/>

</node>

</launch>
```

这是我自己的一个launch文件，分别调用了map_server节点和amcl节点，map_server节点读取了一个我自己之前用gmapping制作的地图（详细教
里 (http://blog.csdn.net/chenxingwangzi/article/details/49802763)），之后调用amcl节点，订阅了scan激光主题，设置了一些参数，参数详细
里 (http://wiki.ros.org/amcl)。
注意一定要将tf tree设置好！！坑了大部分人的都是这个tf。

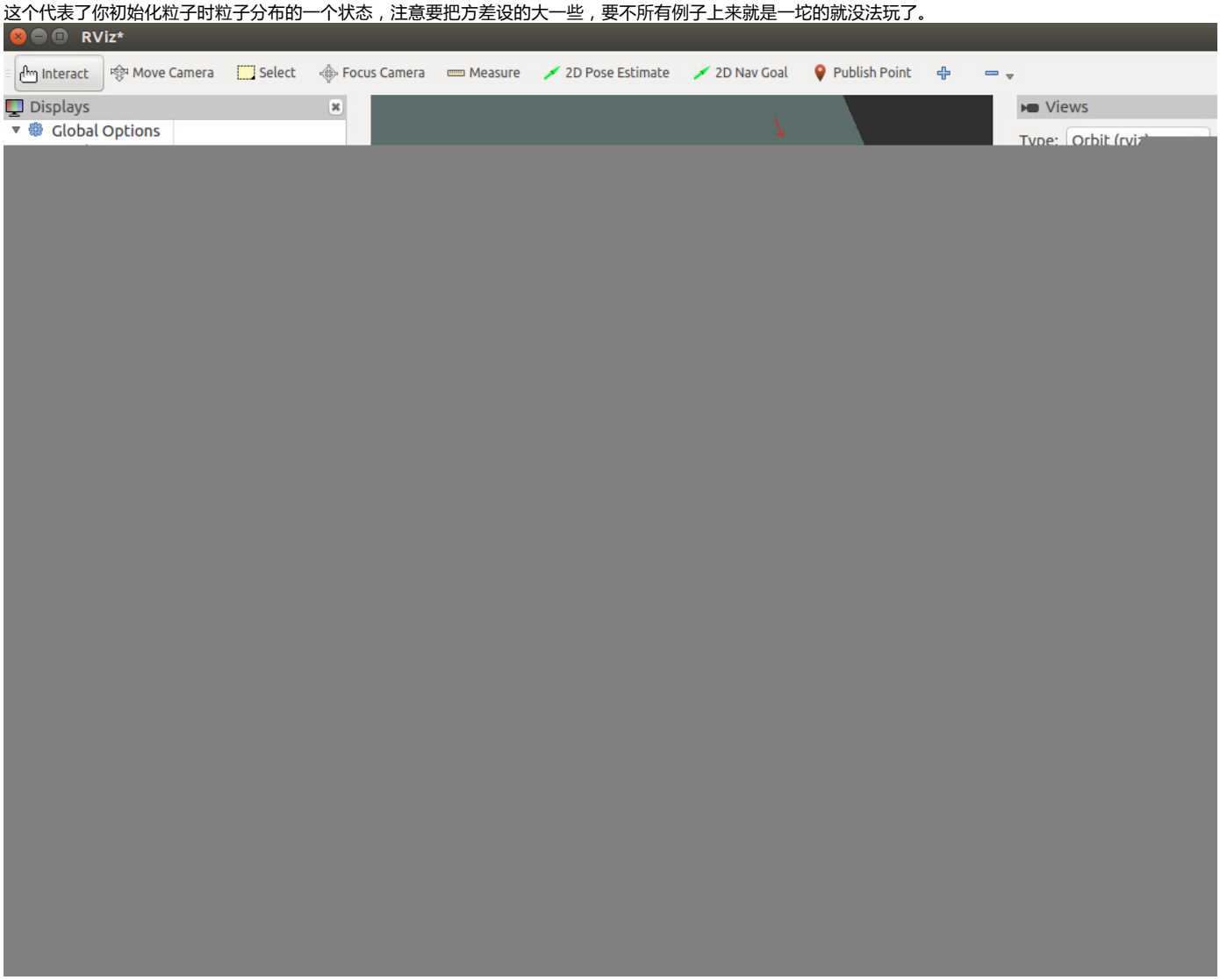


tf显示的就是当前的机器人位姿。
注意：要将rviz的fixed frame设成map，因为map才是global_frame_id。

2015-11-27增补

今天才发现了rviz居然连粒子都可以显示，显示让我对amcl粒子更新有了更深刻的理解。
首先在参数表里面有几个比较重要的参数。
~initial_pose_x (double, default: 0.0 meters)
Initial pose mean (x), used to initialize filter with Gaussian distribution.
~initial_pose_y (double, default: 0.0 meters)
Initial pose mean (y), used to initialize filter with Gaussian distribution.
~initial_pose_a (double, default: 0.0 radians)
Initial pose mean (yaw), used to initialize filter with Gaussian distribution.
~initial_cov_xx (double, default: 0.5*0.5 meters)
Initial pose covariance (x*x), used to initialize filter with Gaussian distribution.
~initial_cov_yy (double, default: 0.5*0.5 meters)
Initial pose covariance (y*y), used to initialize filter with Gaussian distribution.
~initial_cov_aa (double, default: (π/12)*(π/12) radian)
Initial pose covariance (yaw*yaw), used to initialize filter with Gaussian distribution.

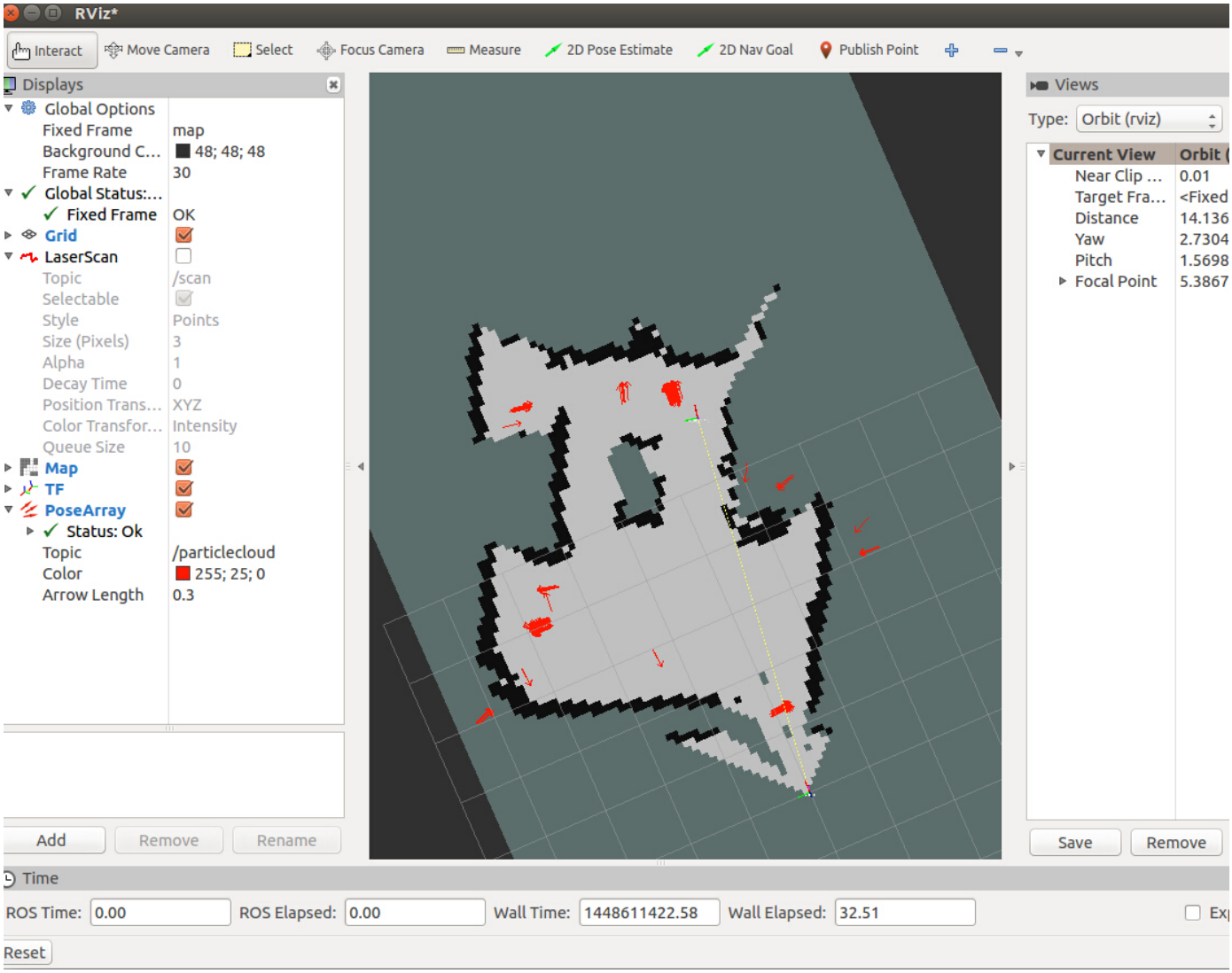
这个代表了你初始化粒子时粒子分布的一个状态，注意要把方差设的大一些，要不所有例子上来就是一坨的就没法玩了。



上图是粒子的一个初始状态~我设的方差比较大~所以分布很大。

(/article/addartic

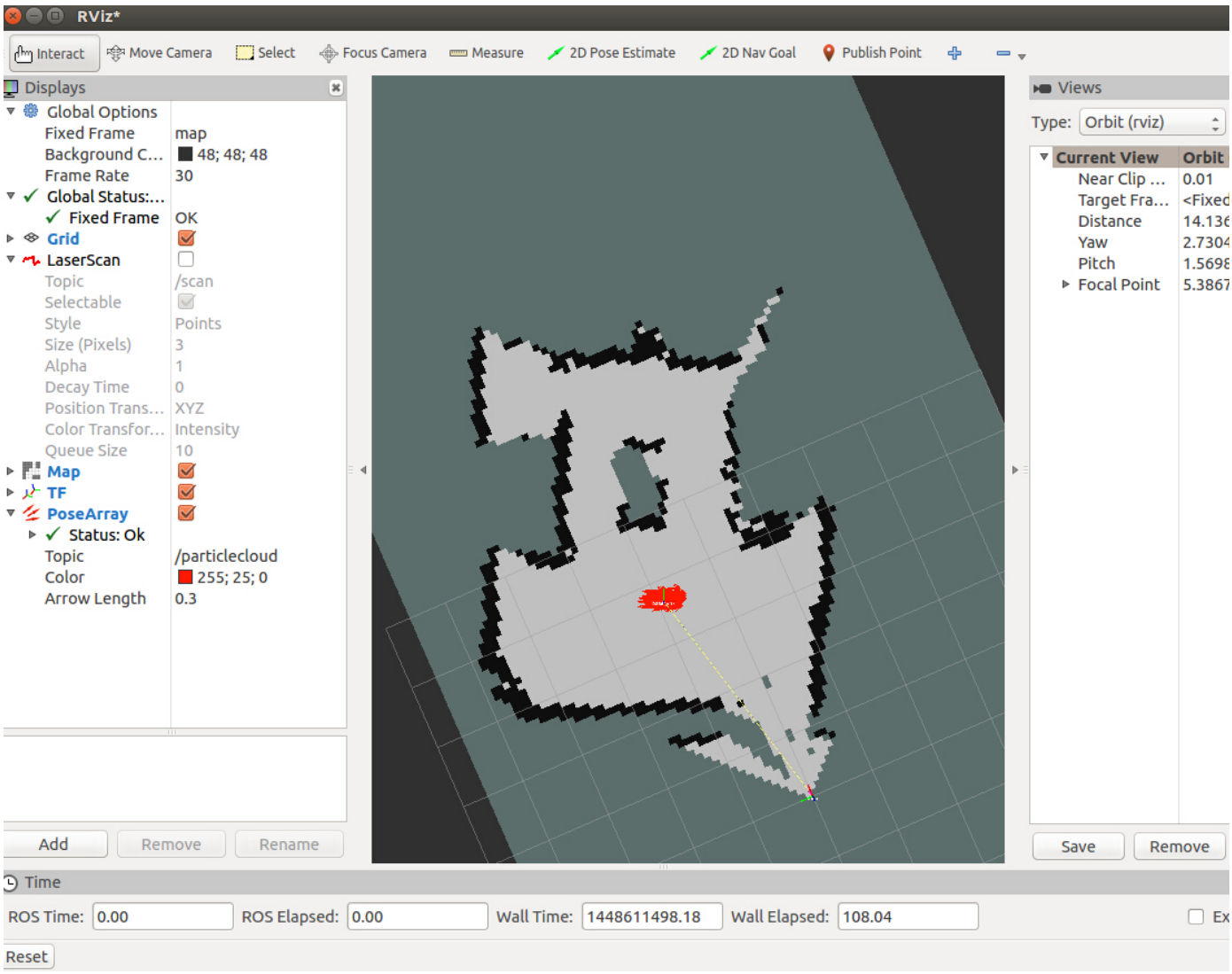
(/article/addforu



上图是更新了一段时间之后~粒子逐渐趋于稳定

(/article/addartic

(/article/addforu



上图为最终状态，趋于稳定

标签：(/tag/)

评论

请登录 (/User/login)后再评论!

(/article/addartic

(/article/addforu

全部评论

目前没有评论