VKBOT智能复合机器人

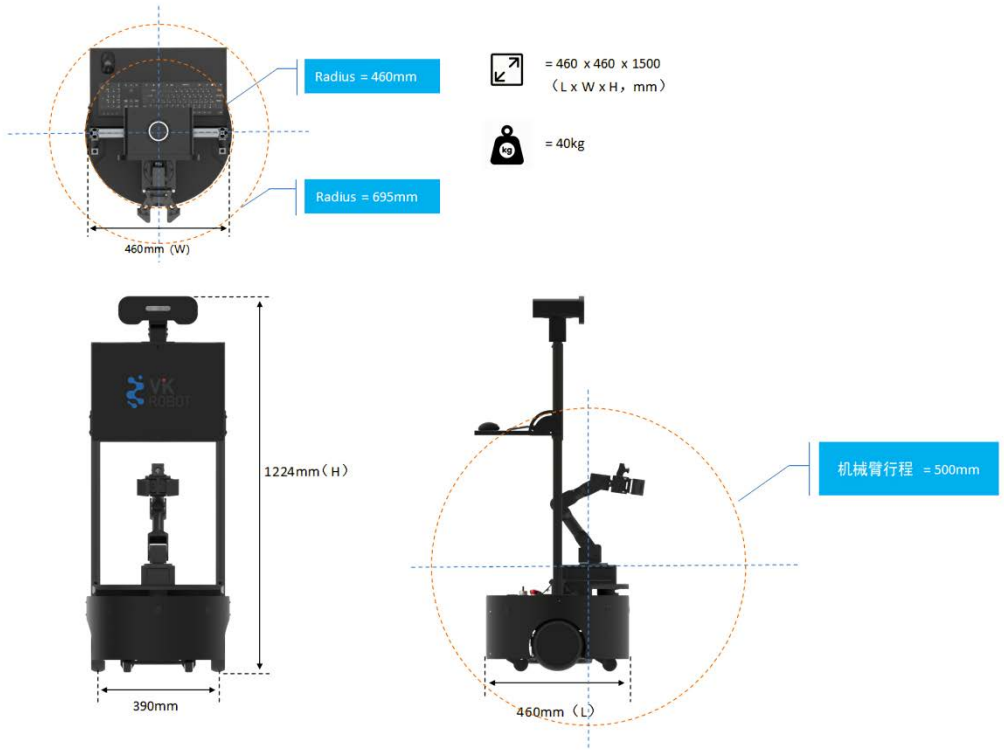
XXX
广州威控机器人研发工程师
XXX@vk-robot.com

ViK ROBOT

目录

1. 整机认识

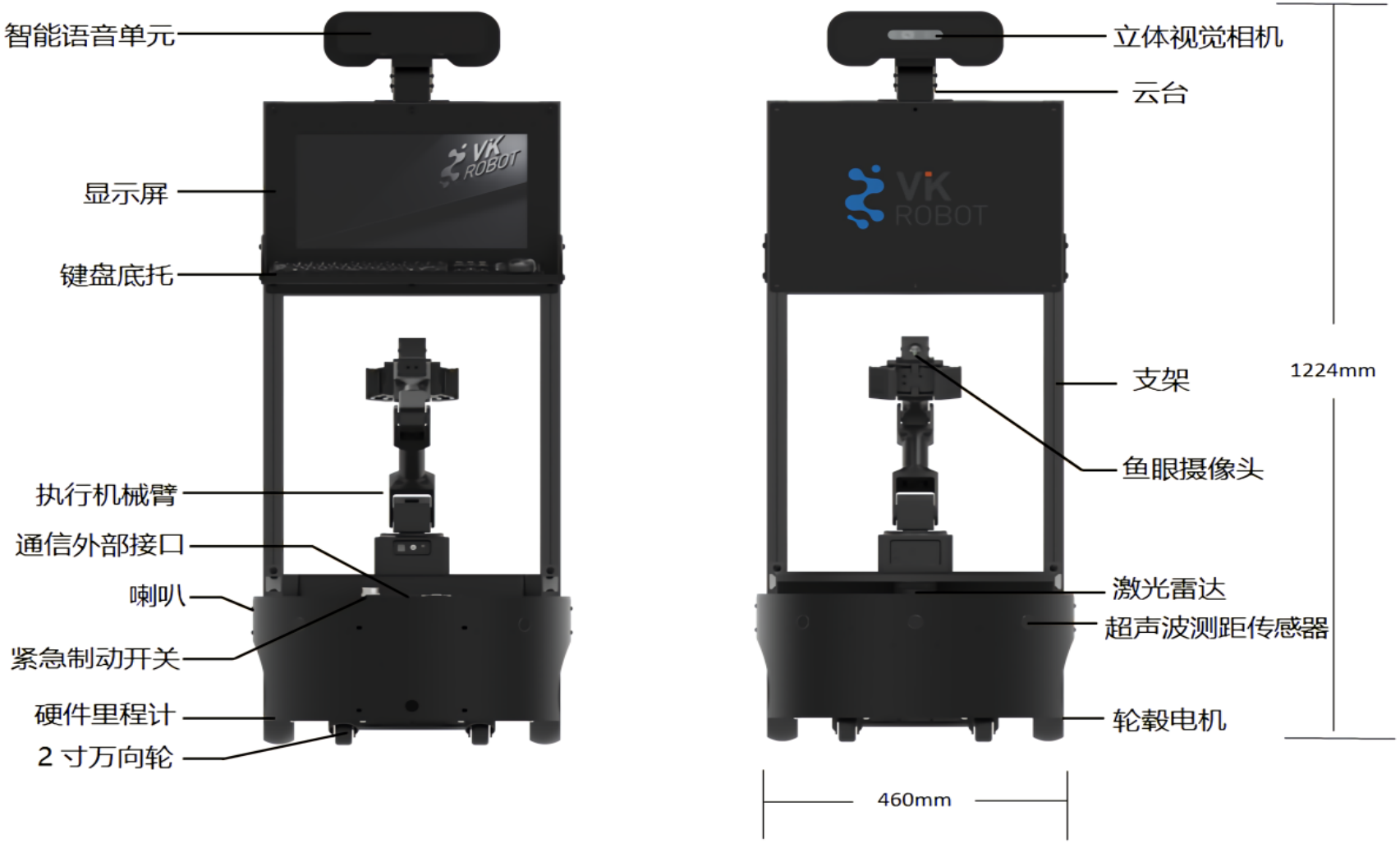
VKBOT智能复合机器人，是集合了威控公司多年的机器人开发经验，打造的一款针对智能机器人教学及相关应用开发的综合性实践平台，该平台具备高精度SLAM定位导航、机械臂视觉抓取、路径规划、物体识别、语音语义识别、人脸人体追踪等功能，学校教学实训、实验室研究开发、技能竞赛参赛训练的理想平台。



针对移动机器人的基础技术，提供运动控制、机器人操作系统、机器人仿真、智能应用等丰富的多元化实践教学案例

- □ 机电系统拆装实践
- □ 电机控制实践
- □ 运动控制实践
- □ 传感器融合实践
- □ Gazebo 仿真实践
- □ 机器人操作系统 ROS 实践
- □ 机械臂应用实践
- □ SLAM 定位导航实践
- □ 路径规划开发实践
- □ 语音语义识别实践
- □ OpenCV 图像应用实践
- □ 深度学习应用实践

1. 整机认识


智能语音单元
显示屏
键盘底托
执行机械臂
通信外部接口
喇叭
紧急制动开关
硬件里程计
2 寸万向轮
立体视觉相机
云台
支架
鱼眼摄像头
激光雷达
超声波测距传感器
轮毂电机
1224mm
460mm

1. 整机认识

底盘参数	操作系统	Linux Ubuntu 16.04 + ROS kinetic
	主控器	NUC8i7BEH
	电机模块	伺服轮毂电机
	轮子	2个主动轮+2个从动轮
	电池	24V 10AH 锂离子电池
	运行速度	< 1.0 m/s
	续航时间	约6个小时
	越障高度	10 mm
	爬波能力	< 15 度
	激光雷达	高精度激光雷达
	视觉传感器	鱼眼摄像
	避障传感器	5路超声波传感器
	开发工具	显示器+无线鼠标键盘

机械臂参数	关节单元	伺服执行单元4+1
	输入电压	12 V
	自由度	5（4 DOF + 1DOF gripper）
	负载	500 g
	工作行程	伸展500mm升降280mm
	重复精度	< 1 mm
	关节速度	46 RPM
	夹持器行程	50 mm
	体积	460 x 460 x1300 mm
结构参数	万向轮	2寸橡胶轮
	驱动轮	6.5寸橡胶轮
	主结构材料	钢板材+铝型材+碳纤维
	载重	50 kg
	自重	40 kg

1. 整机认识

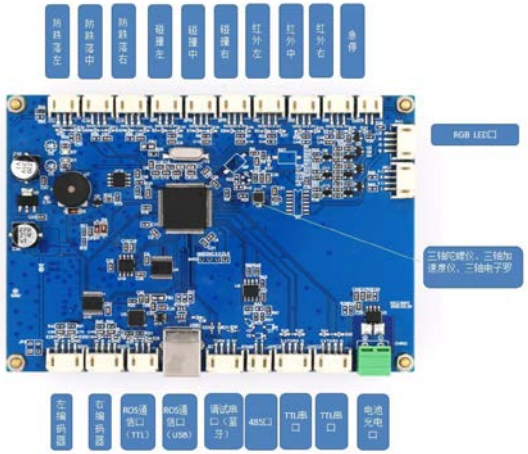
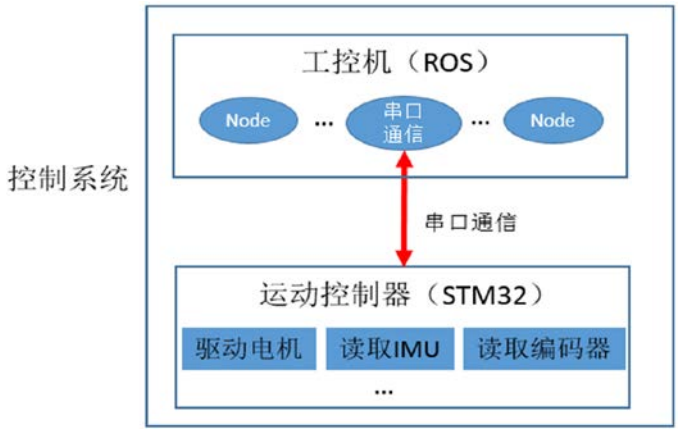


图 1-6 运动控制器

运动控制器

工控机-----Intel NUC

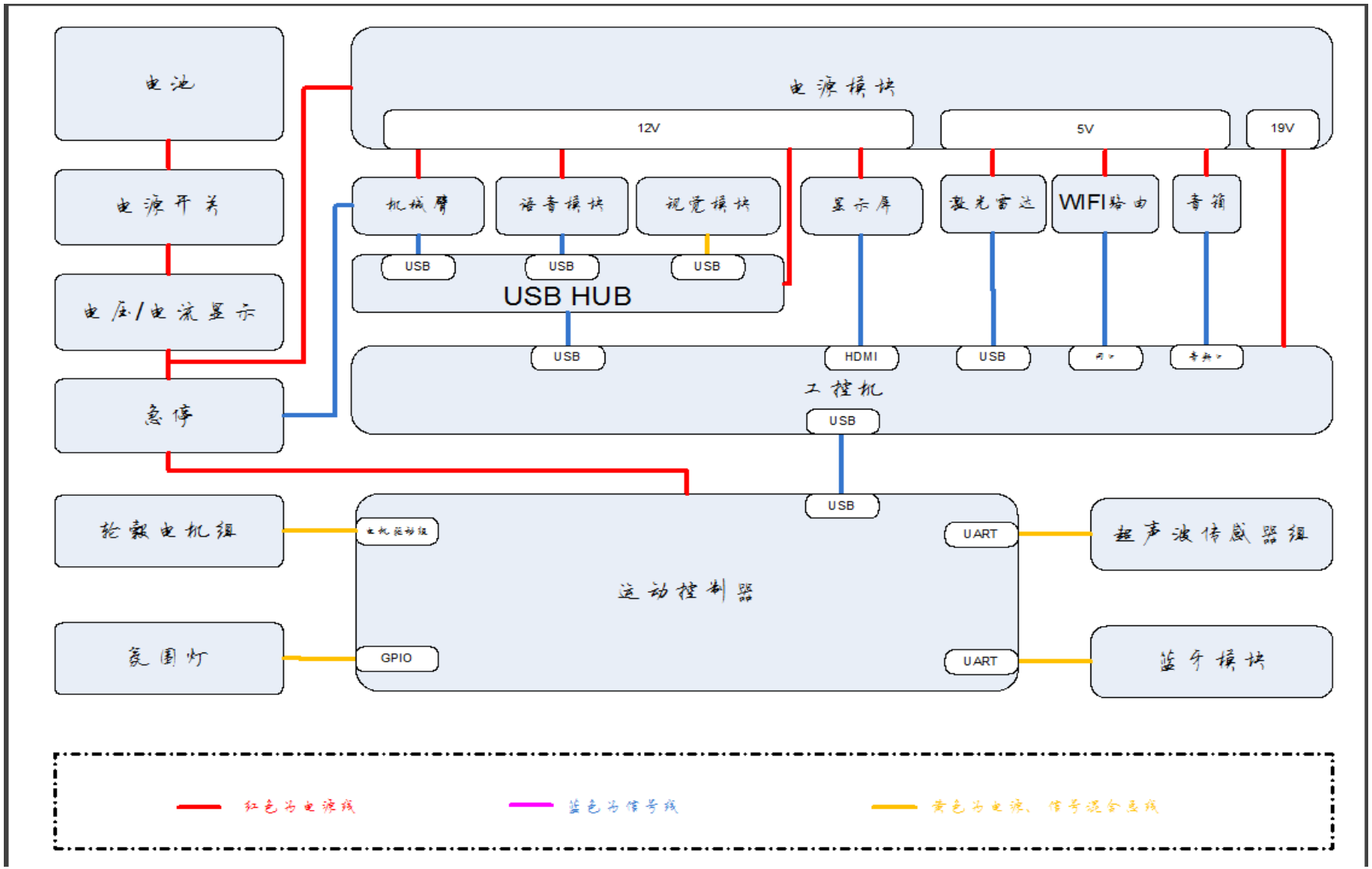


激光雷达

深度摄像头

超声波

1. 整机认识


电池
电源开关
电压/电流显示
急停
电源模块
12V
5V
19V
机械臂
语音模块
视觉模块
显示屏
激光雷达
WIFI路由
音箱
USB
USB
USB
USB HUB
USB
HDMI
USB
工控机
USB
USB
轮毂电机组
运动控制器
UART
超声波传感器组
氛围灯
GPIO
UART
蓝牙模块
红色为电源线
蓝色为信号线
黄色为电源、信号混合总线

1. 整机认识

VKBOT开机流程

- 检查急停开关
- 旋转开关旋钮
- 检查电源电压
- 等待VKBOT启动

系统用户	vkrobot
用户密码	vkrobot
操作系统	Ubuntu 16.04
ROS版本	Kinetic

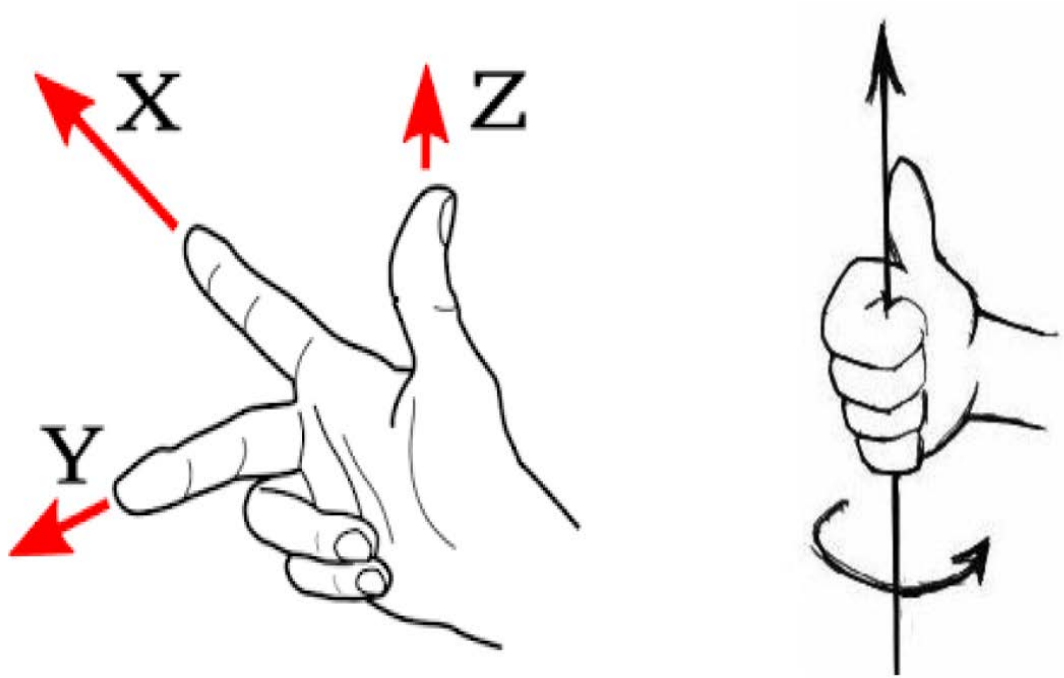
1. 整机认识

工程路径/home/vkbot/vkbot_ws

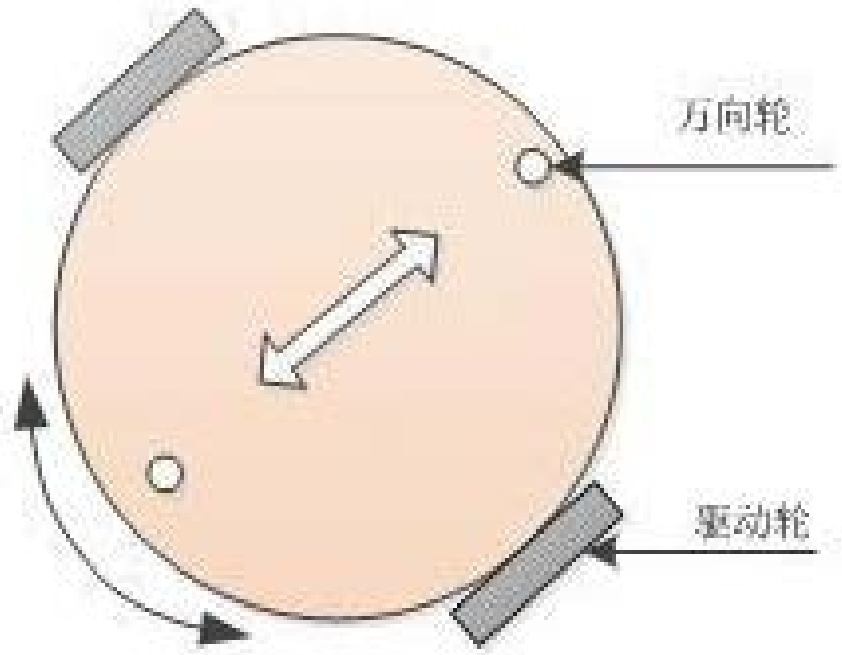
vkbot_driver/kobuki	底盘运动驱动实现
dynamic_reconfigure-kinetic-devel	Realsense深度相机驱动
realsense-ros-development	Realsense深度相机
ydlidar_v1.3.1	激光雷达功能包
vkbot_bringup	启动文件
vkbot_description	vkbot的urdf文件
vkbot_teleop	vkbot键盘控制
darknet_ros	物体识别

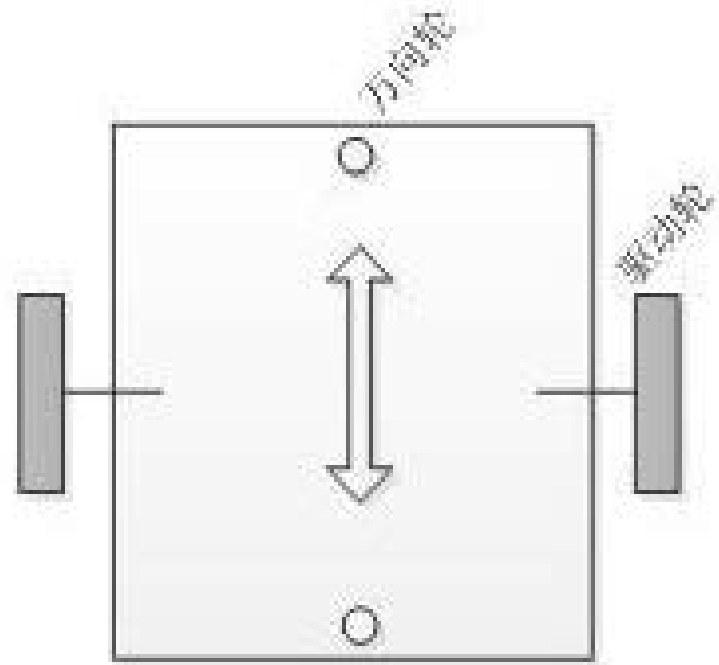
vkbot_add_waypoint	多点导航RVIZ插件
vkbot_follow_waypoints	多点导航功能包
vkbot_navigation	导航功能包
vkbot_simulator	仿真功能包
vkbot_vision	视觉应用相关功能包
vkbot_voice	语音识别功能包

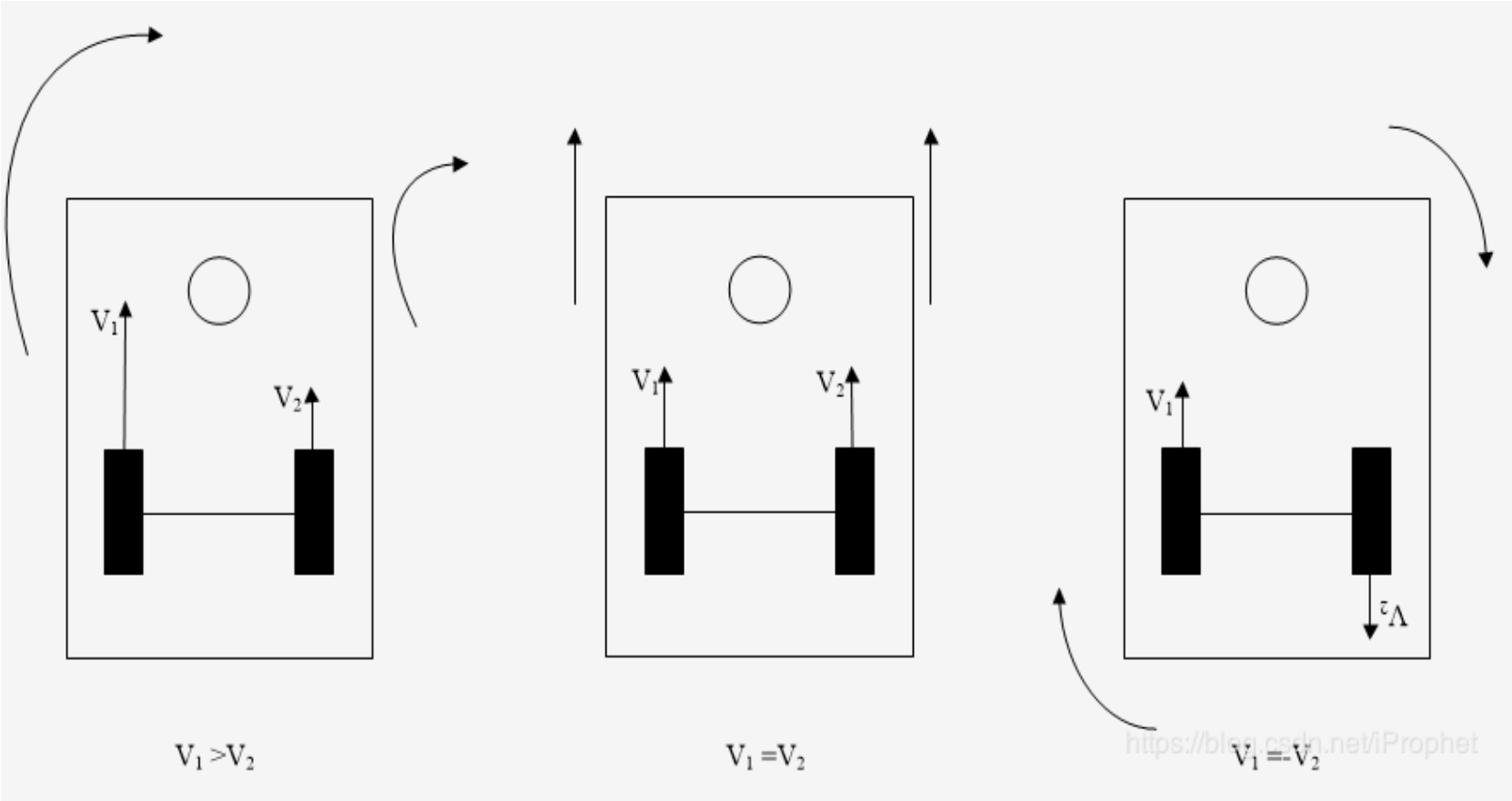
机器人的前方为X轴的正方向 机器人的水平左方向为Y轴正方向 机器人的上方为Z轴正方向



坐标系基础

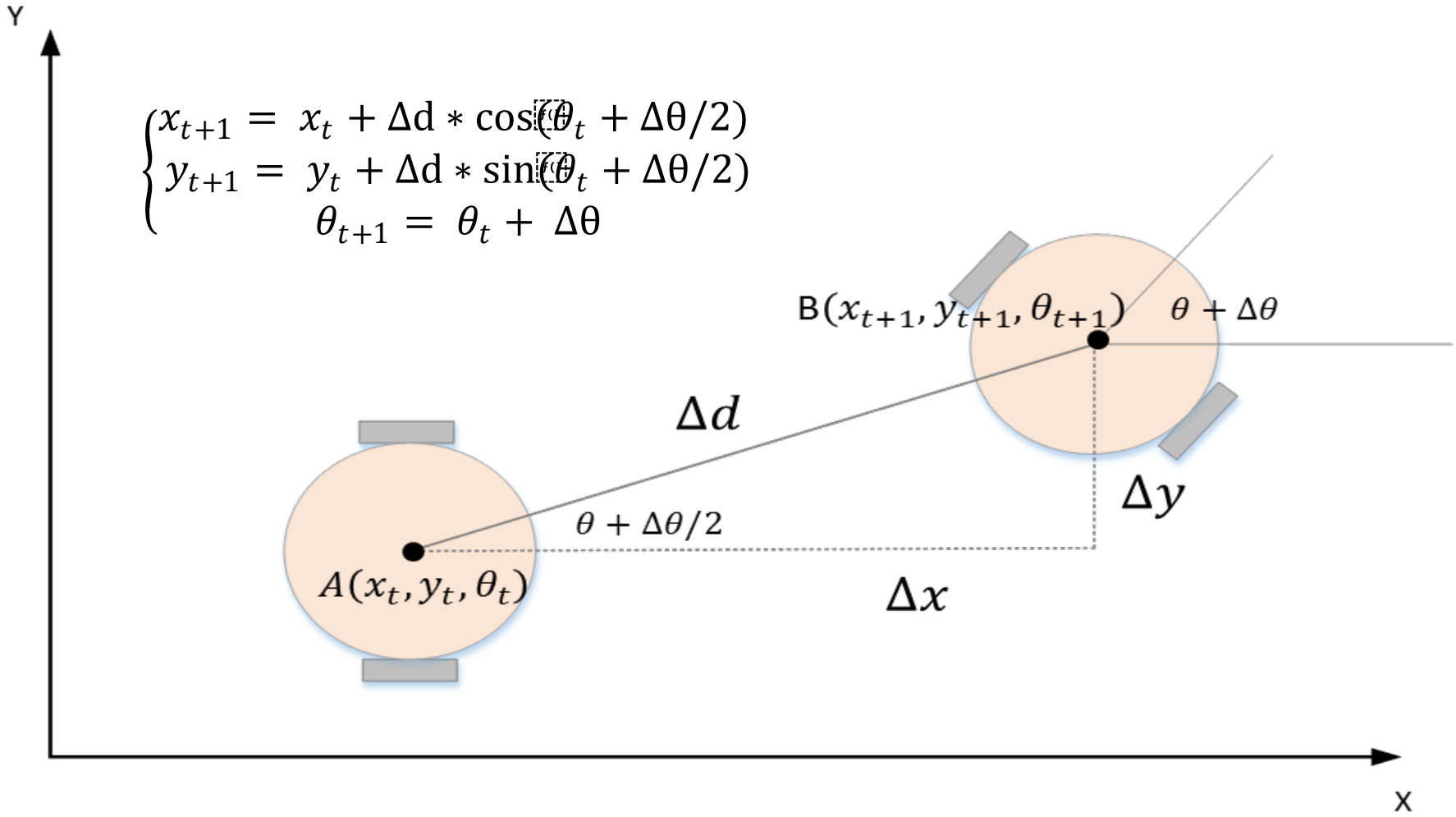

万向轮
驱动轮


万向轮
驱动轮



三种运动状态

$$\begin{cases} x_{t+1} = x_t + \Delta d * \cos(\theta_t + \Delta\theta/2) \\ y_{t+1} = y_t + \Delta d * \sin(\theta_t + \Delta\theta/2) \\ \theta_{t+1} = \theta_t + \Delta\theta \end{cases}$$



VKBOT移动

$ roslaunch vkbot_bringup minimal.launch

```
vkrobot@vkrobot-desktop:~$ roslaunch vkbot_bringup minimal.launch
... logging to /home/vkrobot/.ros/log/628cef94-caf3-11ea-aca9-1c697a621449/roslaunch-vkrobot-desktop-8552.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://vkrobot-desktop:46881/

SUMMARY
========

PARAMETERS
 * /base: kobuki
 * /bumper2pointcloud/pointcloud_radius: 0.24
 * /cmd_vel_mux/yaml_cfg_file: /home/vkrobot/vkb...
 * /diagnostic_aggregator/analyzers/input_ports/contains: ['Digital Input',...
 * /diagnostic_aggregator/analyzers/input_ports/path: Input Ports
 * /diagnostic_aggregator/analyzers/input_ports/remove_prefix: mobile_base_nodel...
 * /diagnostic_aggregator/analyzers/input_ports/timeout: 5.0
 * /diagnostic_aggregator/analyzers/input_ports/type: diagnostic_aggreg...
 * /diagnostic_aggregator/analyzers/kobuki/contains: ['Watchdog', 'Mot...
 * /diagnostic_aggregator/analyzers/kobuki/path: Kobuki
 * /diagnostic_aggregator/analyzers/kobuki/remove_prefix: mobile_base_nodel...
 * /diagnostic_aggregator/analyzers/kobuki/timeout: 5.0
 * /diagnostic_aggregator/analyzers/kobuki/type: diagnostic_aggreg...
 * /diagnostic_aggregator/analyzers/power/contains: ['Battery', 'Lapt...
 * /diagnostic_aggregator/analyzers/power/path: Power System
 * /diagnostic_aggregator/analyzers/power/remove_prefix: mobile_base_nodel...
 * /diagnostic_aggregator/analyzers/power/timeout: 5.0
 * /diagnostic_aggregator/analyzers/power/type: diagnostic_aggreg...
 * /diagnostic_aggregator/analyzers/sensors/contains: ['Cliff Sensor', ...
 * /diagnostic_aggregator/analyzers/sensors/path: Sensors
 * /diagnostic_aggregator/analyzers/sensors/remove_prefix: mobile_base_nodel...
 * /diagnostic_aggregator/analyzers/sensors/timeout: 5.0
```

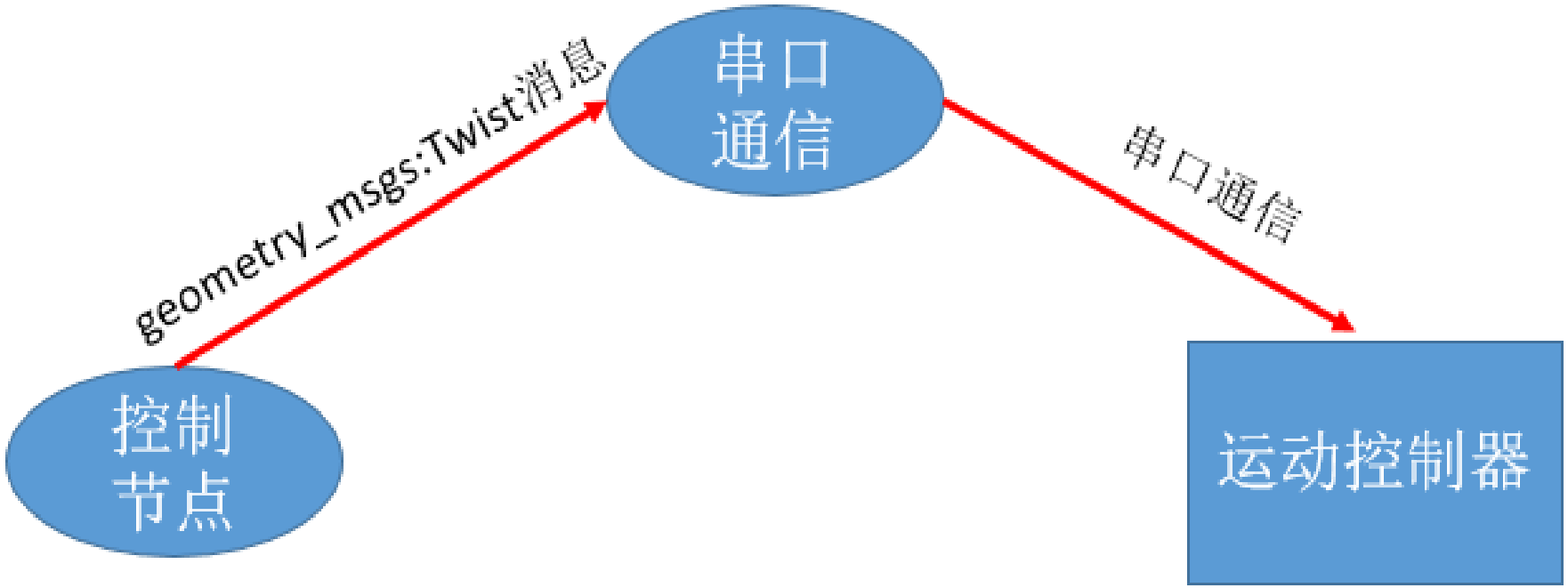
VK ROBOT

$ roslaunch vkbot_teleop keyboard_teleop.launch

```
vkrobot@vkrobot-desktop:~$ roslaunch vkbot_teleop keyboard_teleop.launch
... logging to /home/vkrobot/.ros/log/628cef94-caf3-11ea-aca9-1c697a621449/roslaunch-vkrobot-desktop-9337.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://vkrobot-desktop:45743/

SUMMARY
========

PARAMETERS
 * /rosdistro: kinetic
 * /rosversion: 1.12.14
 * /vkbot_teleop_keyboard/scale_angular: 0.1
 * /vkbot_teleop_keyboard/scale_linear: 0.5

NODES
  /
    vkbot_teleop_keyboard (vkbot_teleop/vkbot_teleop_key)

ROS_MASTER_URI=http://localhost:11311

process[vkbot_teleop_keyboard-1]: started with pid [9376]

Control Your vkbot!
---------------------------
Moving around:
   u    i    o
   j    k    l
   m    ,    .

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
space key, k : force stop
anything else : stop smoothly

CTRL-C to quit

currently:      speed 0.35      turn 0.8
```

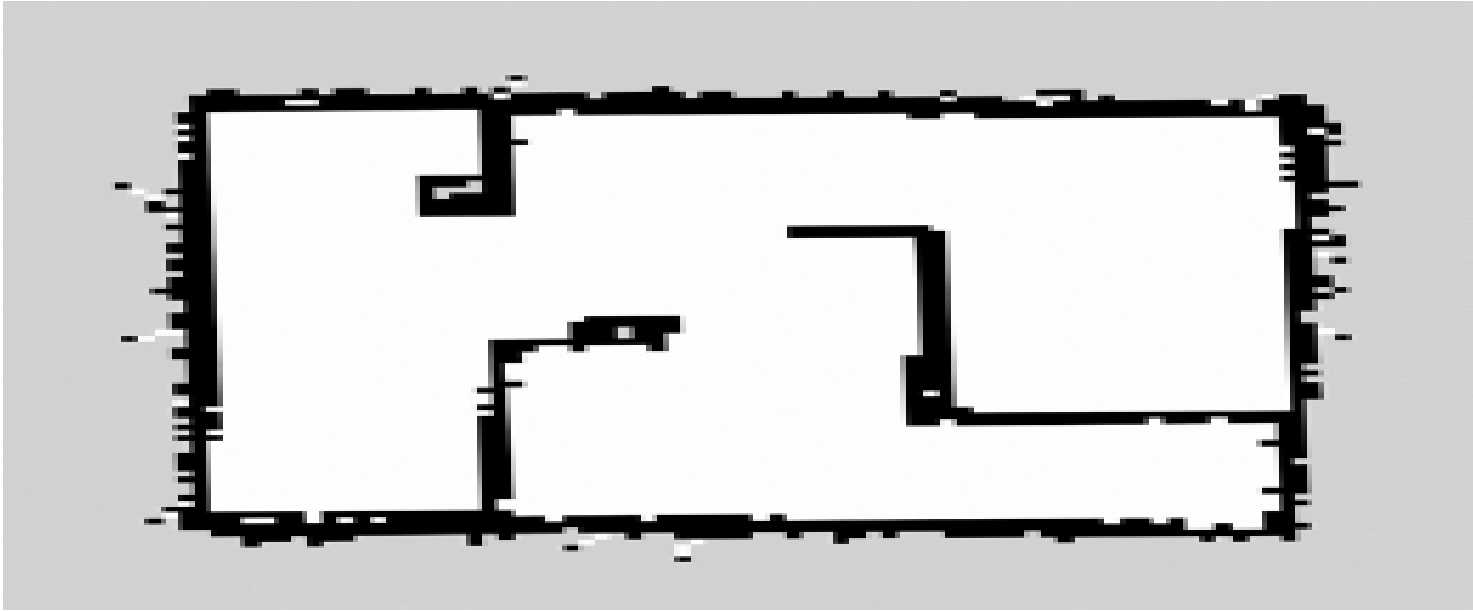
VKBOT移动

启动键盘操纵



SLAM

SLAM（Simultaneous Localization and Mapping,同步定位与建图）
SLAM问题可以描述为: 机器人在未知环境中从一个未知位置开始移动,在移动过程中根据位置估计和地图进行自身定位，同时在自身定位的基础上建造增量式地图，实现机器人的自主定位和导航。

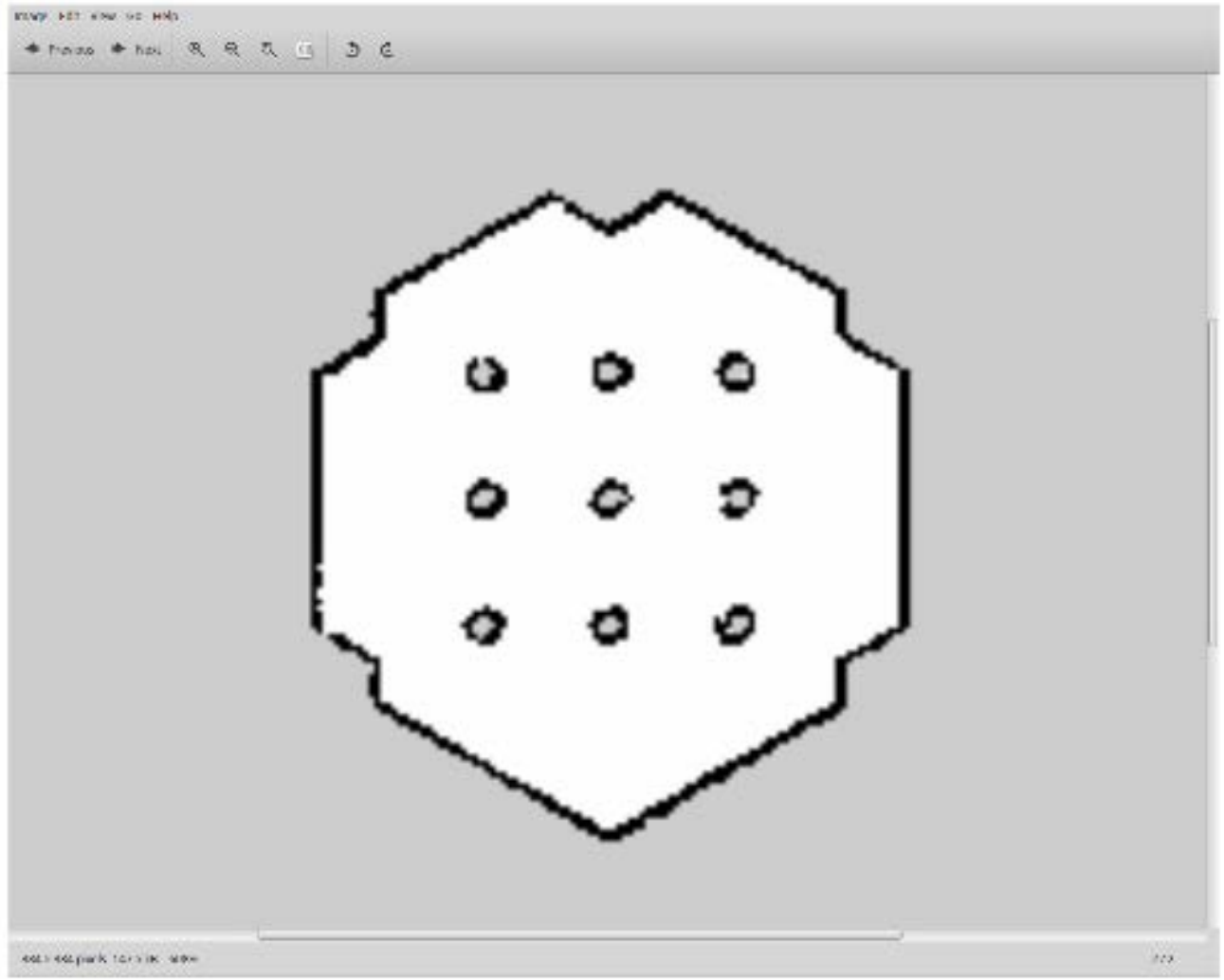


地图

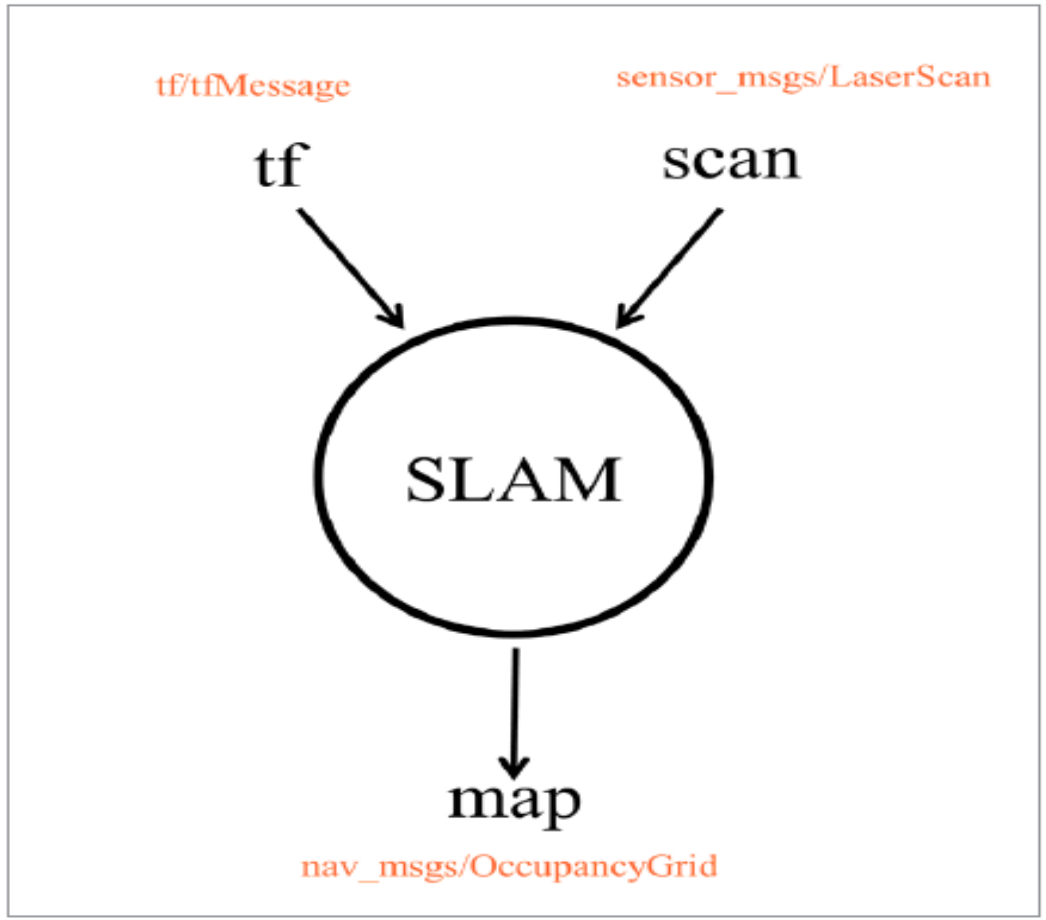
大致可分为三类：栅格表示、几何信息表示和拓扑图表示。

栅格地图表示法即将整个环境分为若干相同大小的栅格，对于每个栅格各指出其中是否存在障碍物。它的优点在于创建和维护容易，尽量的保留了整个环境的各种信息，可以方便地进行自定位和路径规划。缺点在于：当栅格数量增大时对地图的维护行为将变得困难，同时定位过程中搜索空间很大，如果没有较好的简化算法，实现实时应用比较困难。

地图



SLAM算法



TF：坐标关系

Scan: 激光雷达数据

map: 地图数据

SLAM----Gmapping功能包

gmapping功能包集成了Rao-Blackwellized粒子滤波算法，为开发者隐去了复杂的内部实现。gmapping功能包订阅机器人的深度信息、IMU信息和里程计信息，同时完成一些必要参数的配置，即可创建完成基于概率的二维栅格地图。gmapping功能包基于openslam社区的开源SLAM算法。

安装

```
$ sudo apt-get install ros-kinetic-slam-gmapping
```

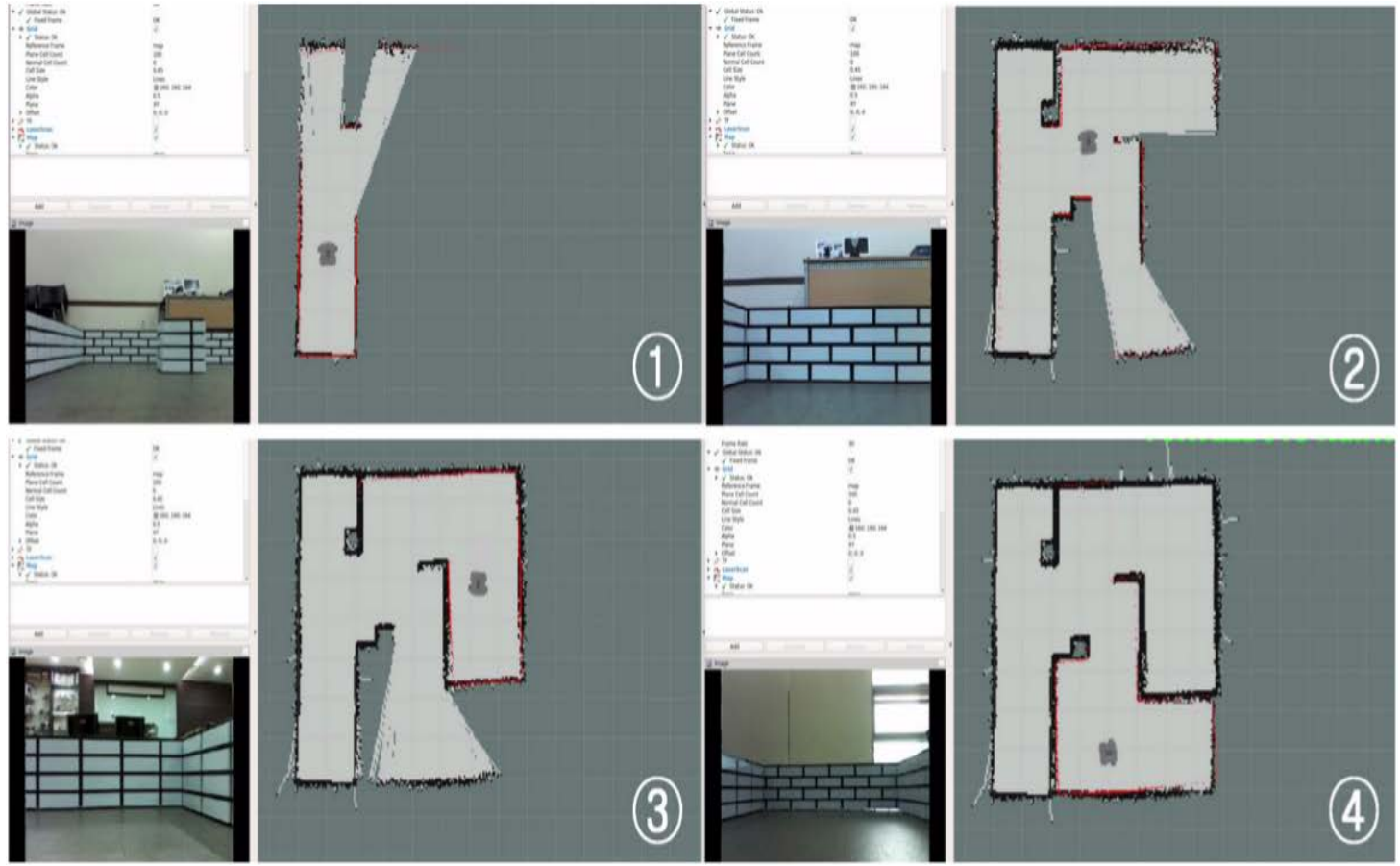
SLAM----Gmapping功能包

参数	类型	默认值	描述
~throttle_scans	int	1	每接收到该数量的激光数据后只处理其中一帧数据
~base_frame	string	base_link	机器人坐标系
~map_frame	string	map	地图坐标系
~odom_frame	string	odom	里程计坐标系
~map_update_interval	float	5.0	地图更新频率
~maxUragne	float	80.0	激光可探测最大距离
~sigma	float	0.05	端点匹配的标准差
~kernelSize	int	1	对应的内核进行查找
~lstep	float	0.05	平移过程中的优化步长
~astep	float	0.05	旋转过程中的优化步长

~iterations	int	5	扫描匹配迭代次数
~lsigma	float	0.075	似然计算的激光标准差
~ogain	float	3.0	似然计算用于平滑重采样效果
~lskip	int	0	每次扫描跳过的光束数
~minimumScore	float	0.0	扫描匹配结果的最低值
~srr	Float	0.1	平移函数，平移时的里程误差
~srt	float	0.2	旋转函数，平移时的里程误差
~str	float	0.1	平移函数，旋转时的里程误差
~stt	float	0.2	旋转函数，旋转时的里程误差
~linearUpdate	float	1.0	机器人每平移该距离处理一次激光
~angularUpdate	float	0.5	机器人每旋转该弧度处理一次激光
~temporalUpdate	float	-1.0	小于零关闭基于时间的更新
~resampleThreshold	float	0.5	基于Neff的重采样阈值

~particles	int	30	滤波器中的粒子数
~xmin	int	-100.0	地图x最小尺寸
~ymin	int	-100.0	地图y最小尺寸
~xmax	int	100.0	地图x最大尺寸
~ymax	int	100.0	地图y最大尺寸
~delta	float	0.05	地图分辨率
~llsamplerange	float	0.01	似然计算的平移采样距离
~llsamplestep	float	0.01	似然计算的平移采样步长
~lasamplerange	float	0.005	似然计算的角度采样距离
~lasamplestep	float	0.005	似然计算的角度采样步长
~transform_publil sh_period	float	0.05	TF变换发布的时间间隔
~occ_thresh	float	0.25	栅格地图占用率阈值
~maxRange	float	-	传感器最大范围

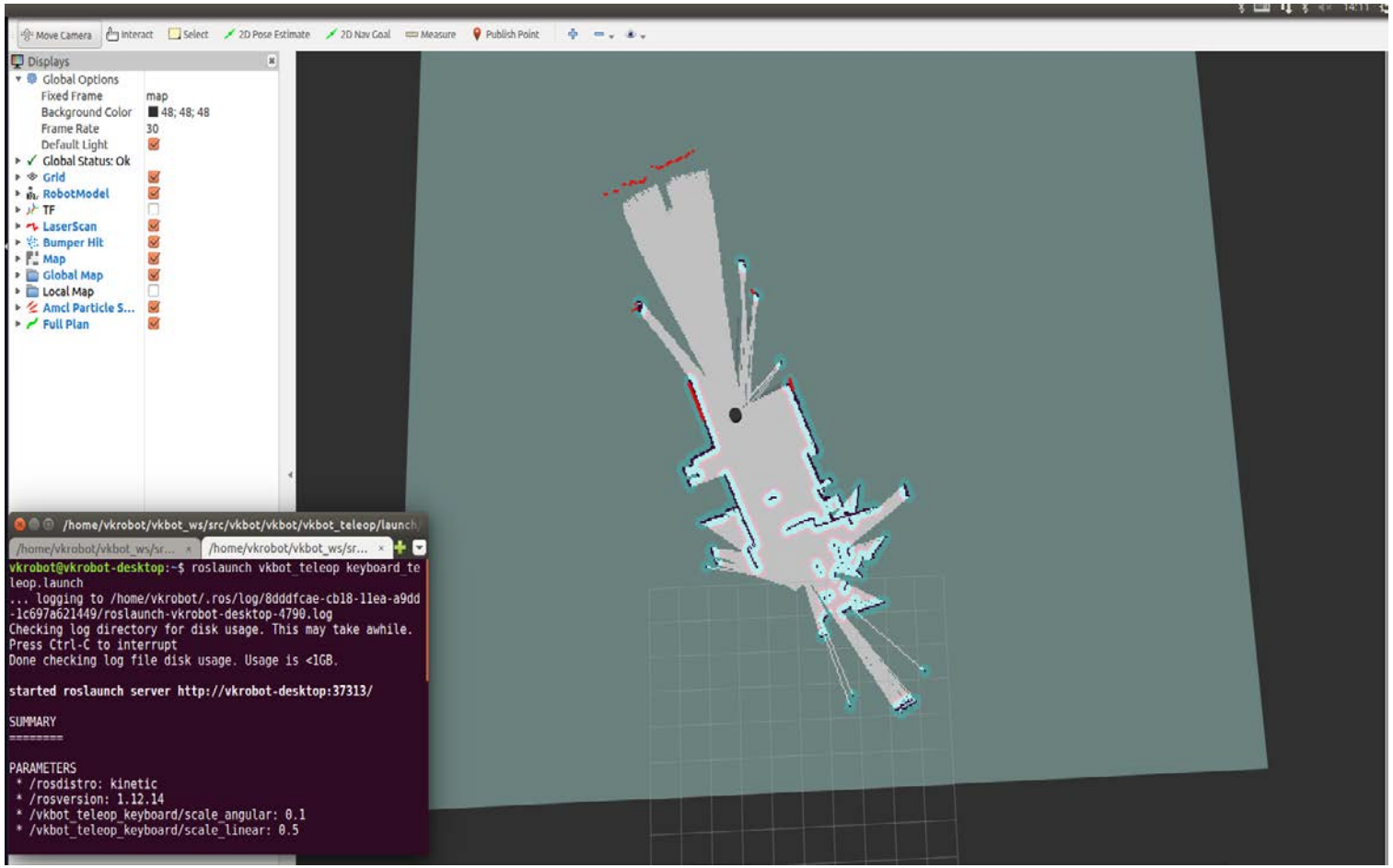
2. SLAM



```
$ roslaunch vkbot_navigation vkbot_gmapping_rviz_demo.launch
```

```
vkrobot@vkrobot-desktop:~$ roslaunch vkbot_navigation vkbot_gmapping_rviz_demo.launch
... logging to /home/vkrobot/.ros/log/19d1de4e-caf4-11ea-aca9-1c697a621449/roslaunch-vkrobot-desktop-9838.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://vkrobot-desktop:34777/

SUMMARY
========

PARAMETERS
 * /base: kobuki
 * /bumper2pointcloud/pointcloud_radius: 0.24
 * /cmd_vel_mux/yaml_cfg_file: /home/vkrobot/vkb...
 * /diagnostic_aggregator/analyzers/input_ports/contains: ['Digital Input',...
 * /diagnostic_aggregator/analyzers/input_ports/path: Input Ports
 * /diagnostic_aggregator/analyzers/input_ports/remove_prefix: mobile_base_nodel...
 * /diagnostic_aggregator/analyzers/input_ports/timeout: 5.0
 * /diagnostic_aggregator/analyzers/input_ports/type: diagnostic_aggreg...
 * /diagnostic_aggregator/analyzers/kobuki/contains: ['Watchdog', 'Mot...
 * /diagnostic_aggregator/analyzers/kobuki/path: Kobuki
 * /diagnostic_aggregator/analyzers/kobuki/remove_prefix: mobile_base_nodel...
 * /diagnostic_aggregator/analyzers/kobuki/timeout: 5.0
 * /diagnostic_aggregator/analyzers/kobuki/type: diagnostic_aggreg...
 * /diagnostic_aggregator/analyzers/power/contains: ['Battery', 'Lapt...
 * /diagnostic_aggregator/analyzers/power/path: Power System
 * /diagnostic_aggregator/analyzers/power/remove_prefix: mobile_base_nodel...
 * /diagnostic_aggregator/analyzers/power/timeout: 5.0
 * /diagnostic_aggregator/analyzers/power/type: diagnostic_aggreg...
 * /diagnostic_aggregator/analyzers/sensors/contains: ['Cliff Sensor', ...
 * /diagnostic_aggregator/analyzers/sensors/path: Sensors
 * /diagnostic_aggregator/analyzers/sensors/remove_prefix: mobile_base_nodel...
 * /diagnostic_aggregator/analyzers/sensors/timeout: 5.0
 * /diagnostic_aggregator/analyzers/sensors/type: diagnostic_aggreg...
 * /diagnostic_aggregator/base_path:
 * /diagnostic_aggregator/pub_rate: 1.0
```

```
$ roslaunch vkbot_teleop keyboard_teleop.launch
```



```
$ roscd vkbot_navigation
$ cd maps
$ rosrun map_server map_saver -f vk_test_2
```

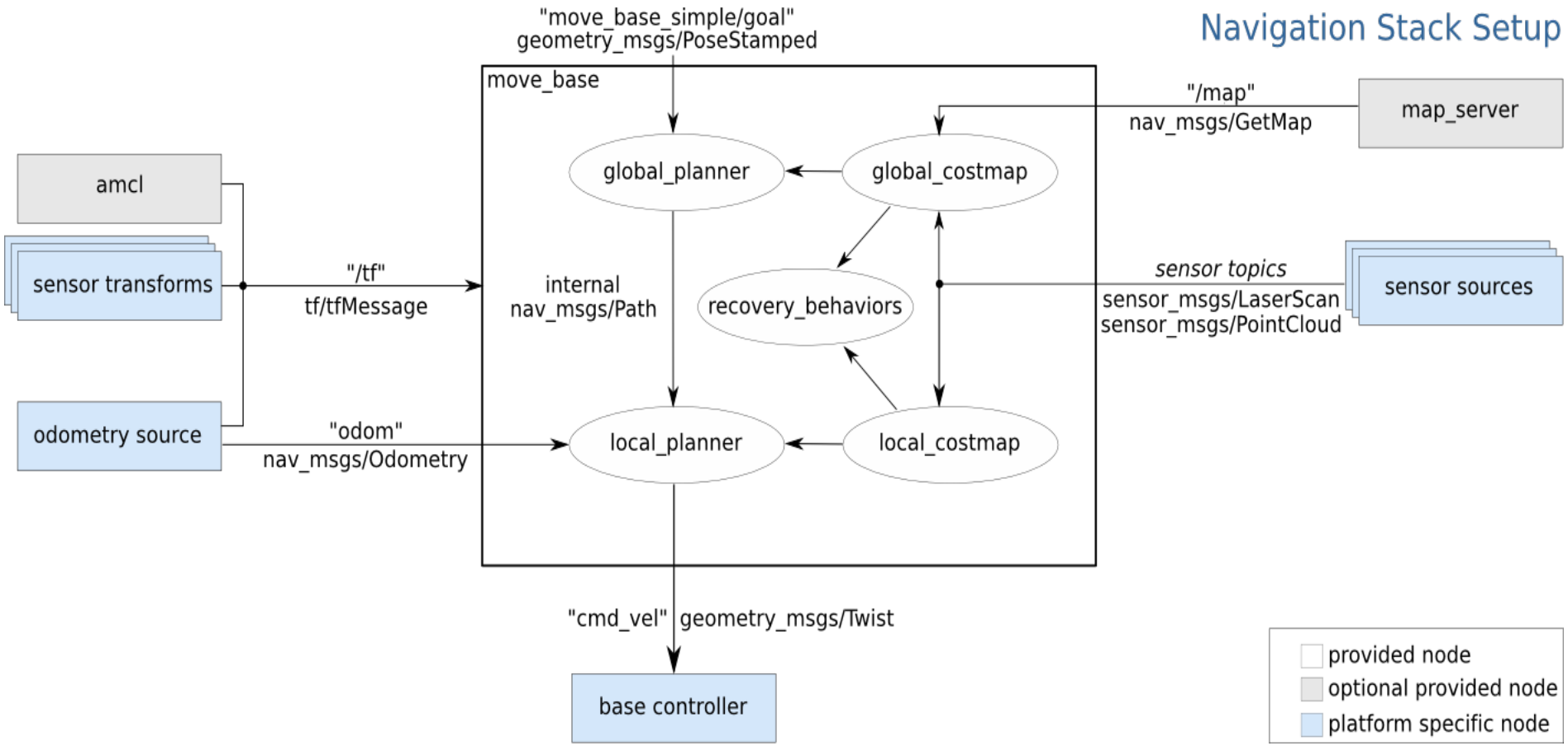
```
vkrobot@vkrobot-desktop:~$ cd vkbot_ws/src/vkbot/vkbot_apps/vkbot_navigation/maps/
vkrobot@vkrobot-desktop:~/vkbot_ws/src/vkbot/vkbot_apps/vkbot_navigation/maps$ rosrun map_server map_saver -f vk_te
st_2
[ INFO] [1595312136.577260847]: Waiting for the map
[ INFO] [1595312136.852730225]: Received a 608 X 576 map @ 0.050 m/pix
[ INFO] [1595312136.852752955]: Writing map occupancy data to vk_test_2.pgm
[ INFO] [1595312136.858657783]: Writing map occupancy data to vk_test_2.yaml
[ INFO] [1595312136.858734413]: Done

vkrobot@vkrobot-desktop:~/vkbot_ws/src/vkbot/vkbot_apps/vkbot_navigation/maps$
```

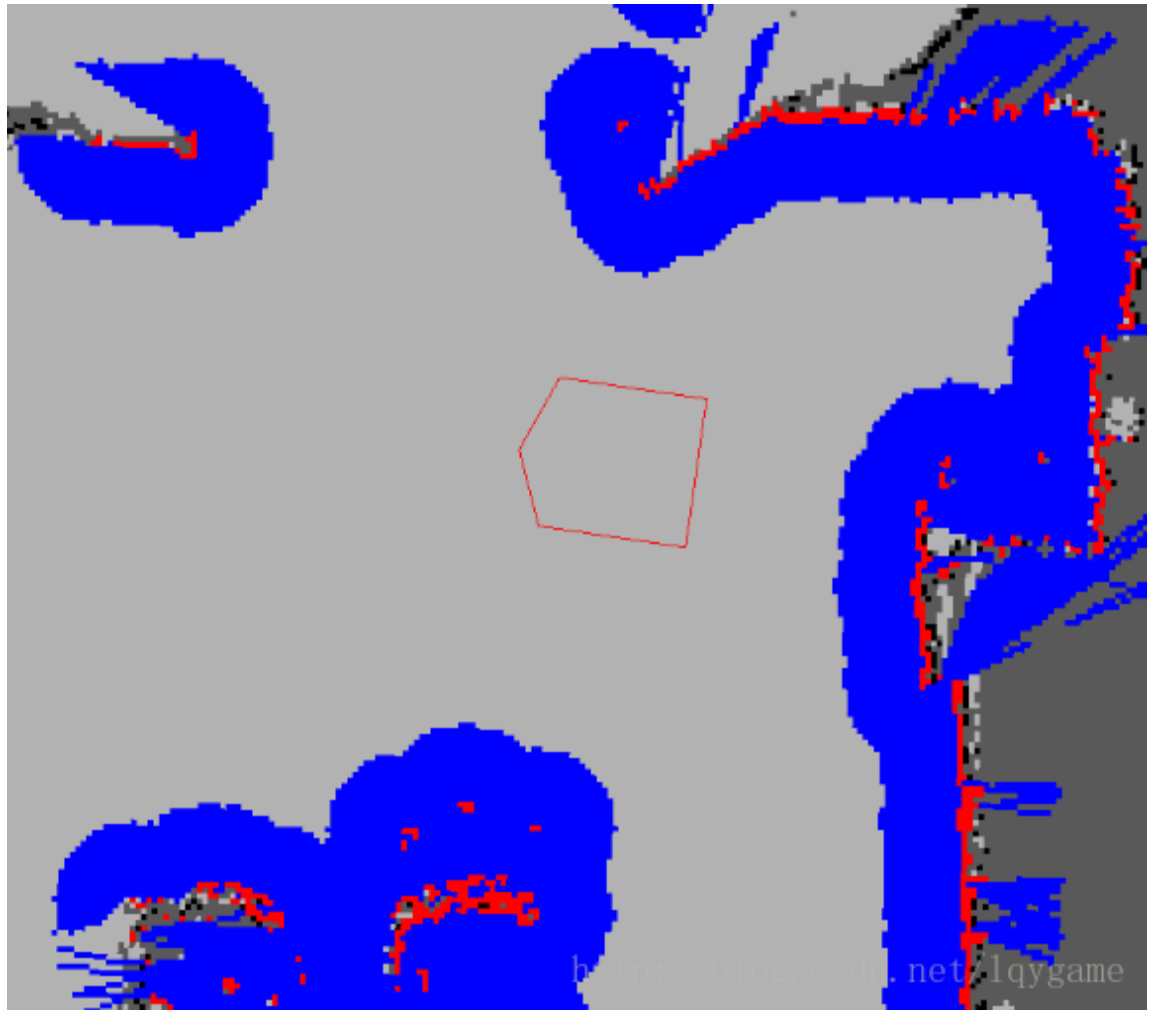
图 保存地图

导航技术

导航技术是移动机器人的一项核心技术之一，它是指移动机器人通过传感器感知环境信息和自身状态，实现在有障碍的环境中面向目标的自主运动。目前，移动机器人主要的导航方式包括：磁导航、惯性导航、视觉导航等。

3. 导航

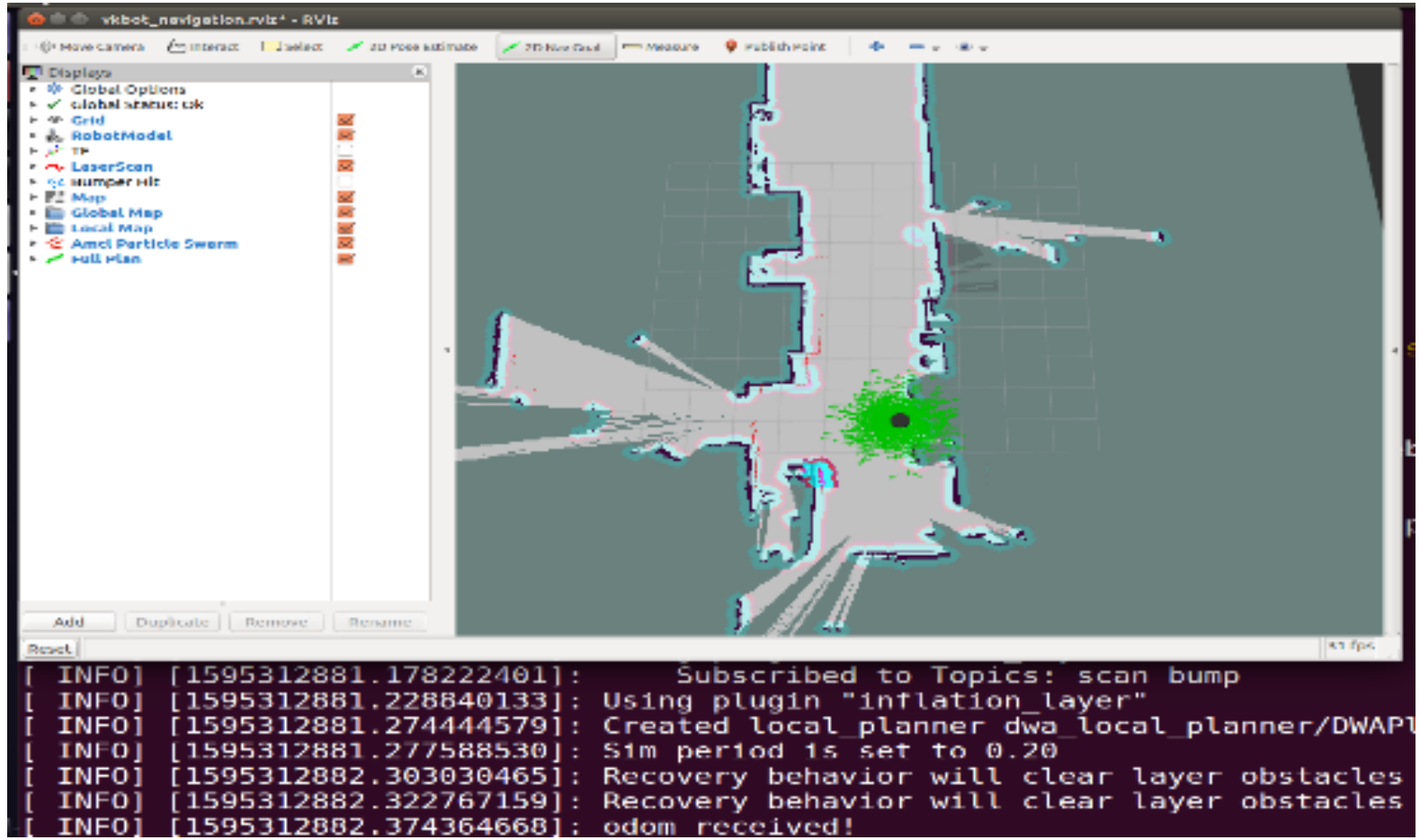

Navigation Stack Setup
"move_base_simple/goal"
geometry_msgs/PoseStamped
move_base
"/map"
nav_msgs/GetMap
map_server
global_planner
global_costmap
amcl
sensor transforms
"/tf"
tf/tfMessage
internal
nav_msgs/Path
recovery_behaviors
sensor topics
sensor_msgs/LaserScan
sensor_msgs/PointCloud
sensor sources
odometry source
"odom"
nav_msgs/Odometry
local_planner
local_costmap
"cmd_vel"
geometry_msgs/Twist
base controller
provided node
optional provided node
platform specific node



```
$ roslaunch vkbot_navigation vkbot_navigation_teb__rviz_test_demo.launch
```

```
<!-- Map server -->
<arg name="map_file" default="$(find vkbot_navigation)/maps/vk_test_2.yaml" />
<node name="map_server" pkg="map_server" type="map_server" args="$(arg map_file)" />

<!-- AMCL -->
<arg name="custom_amcl_launch_file" default="$(find vkbot_navigation)/launch/includes/amcl/amcl.launch.xml"/>
<arg name="initial_pose_x" default="0.0"/> <!-- Use 17.0 for willow's map in simulation -->
<arg name="initial_pose_y" default="0.0"/> <!-- Use 17.0 for willow's map in simulation -->
<arg name="initial_pose_a" default="0.0"/>
<include file="$(arg custom_amcl_launch_file)">
  <arg name="initial_pose_x" value="$(arg initial_pose_x)"/>
  <arg name="initial_pose_y" value="$(arg initial_pose_y)"/>
  <arg name="initial_pose_a" value="$(arg initial_pose_a)"/>s
</include>
```

```
$ roslaunch vkbot_navigation vkbot_navigation_teb__rviz_test_demo.launch
```



估计初始位姿

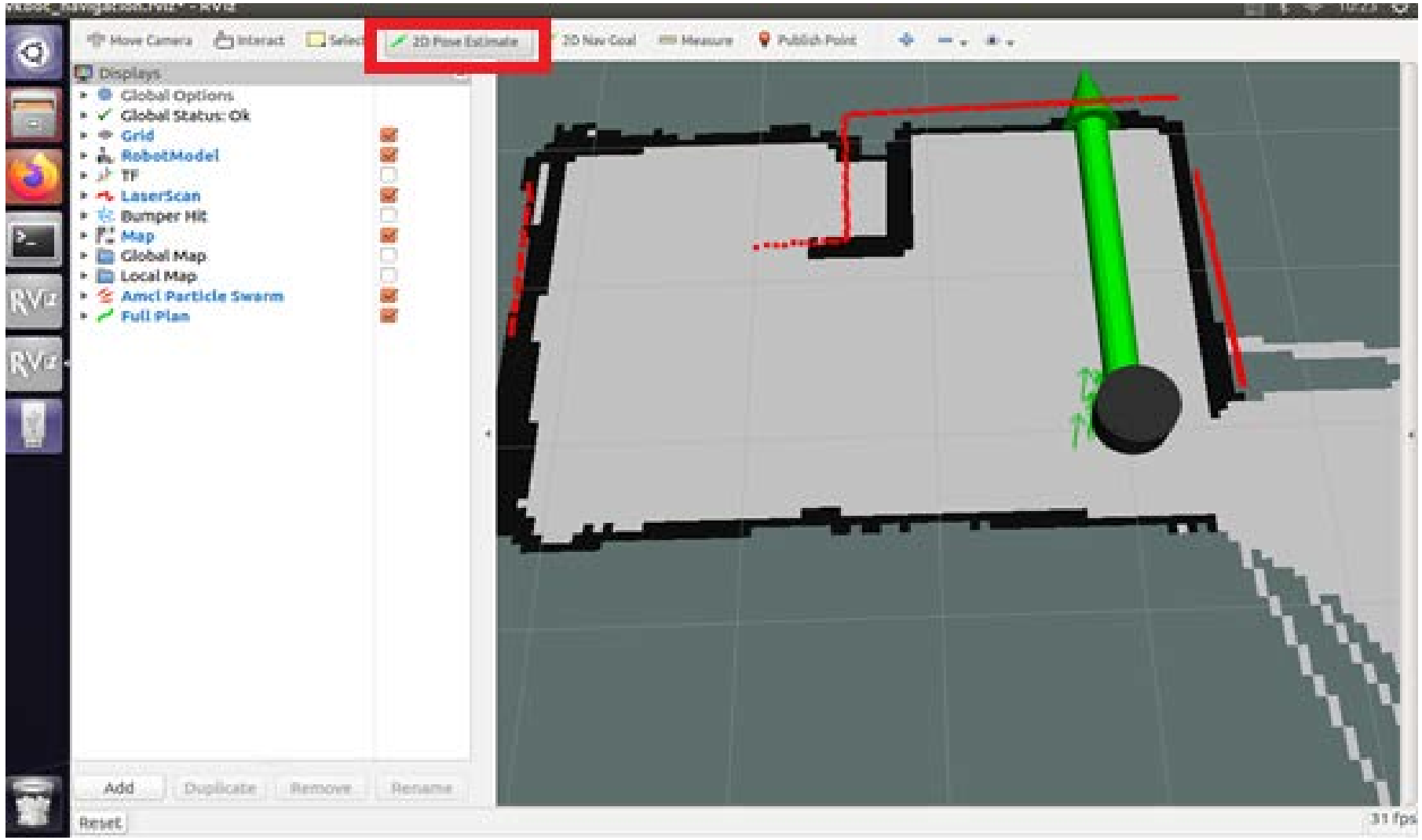
- 点击2D Pose Estimate按钮。
- 单击地图中VKBOT所在的近似点，然后拖动光标以指示VKBOT面向的方向。

  或者可以选择在刚开始的时候，就把VKBOT移动到构建地图的起始位姿上。使得VKBOT在Rviz所显示的位置正确

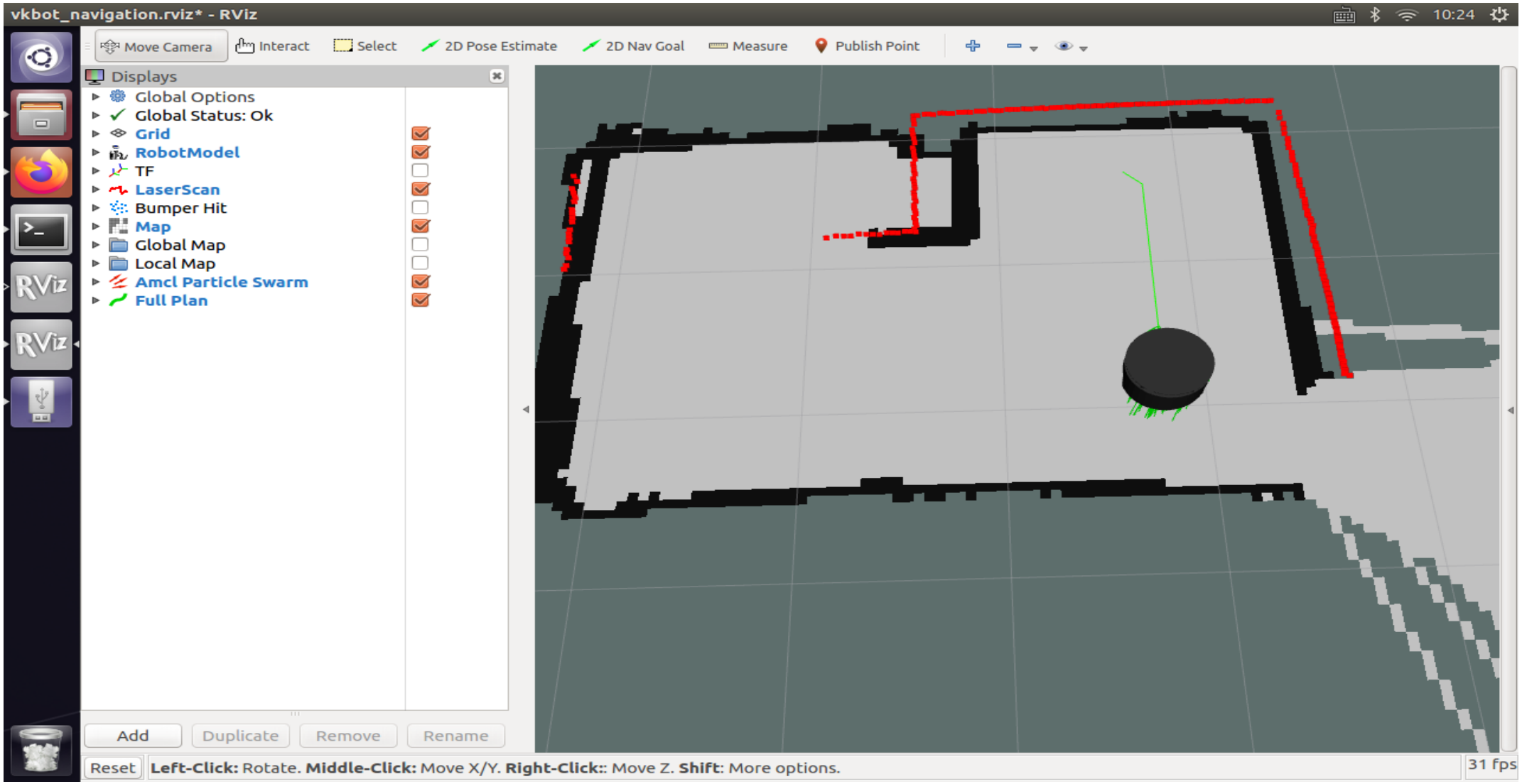
发送导航目标

- 点击2D Nav Goal按钮。
- 单击地图中的特定点以设置目标位置，然后将光标拖到VKBOT最终应面向的方向
- 机器人将根据地图规划一条路径，以避开通往目的地的障碍。然后，机器人沿着路径移动。此时，即使突然检测到障碍物，机器人也会避开障碍物到达目标点。

3. 导航


2D Pose Estimate
Displays
Global Options
Global Status: Ok
Grid
RobotModel
TF
LaserScan
Bumper Hit
Map
Global Map
Local Map
Amcl Particle Swarm
Full Plan
Add
Duplicate
Remove
Rename
Reset

3. 导航


vkbot_navigation.rviz* - RViz
10:24
Move Camera
Interact
Select
2D Pose Estimate
2D Nav Goal
Measure
Publish Point
Displays
Global Options
Global Status: Ok
Grid
RobotModel
TF
LaserScan
Bumper Hit
Map
Global Map
Local Map
Amcl Particle Swarm
Full Plan
Add
Duplicate
Remove
Rename
Reset
Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click:: Move Z. Shift: More options.
31 fps

多点导航技术要求

（1）PYTHON编程基础

（2）机器人操作系统基础

（3）导航功能包move_base

（4）动作通信机制

（5）机器人坐标变换

（6）坐标变换功能包TF

```python
#!/usr/bin/env python
import rospy
import actionlib
from move_base_msgs.msg import MoveBaseAction, MoveBaseGoal
from actionlib_msgs.msg import GoalStatus
#GOalStatus SUCCEEDED=3
import tf

waypoints=[
    [(3.5,2.5,0.0),(0.0,0.0,1.57)],

    [(3.5,-2.5,0.0),(0.0,0.0,3.14)]
]

```

PYTHON多点导航脚本

（1）导入需要的模块rospy等

（2）定义导航点列表

3. 导航

坐标转换函数

导航点格式

位置: x, y,z　　单位：米

欧拉角：x,y,z　　单位：弧度

导航目标：四元数

节点关闭函数

日志记录

取消上一次的目标

```python
def goal_pose(pose):
    goal = MoveBaseGoal()
    goal.target_pose.header.frame_id="map"
    goal.target_pose.pose.position.x = pose[0][0]
    goal.target_pose.pose.position.y = pose[0][1]
    goal.target_pose.pose.position.z = pose[0][2]

    x, y, z, w = tf.transformations.quaternion_from_euler\
        (pose[1][0],pose[1][1],pose[1][2])

    goal.target_pose.pose.orientation.x = x
    goal.target_pose.pose.orientation.y = y
    goal.target_pose.pose.orientation.z = z
    goal.target_pose.pose.orientation.w = w
    return goal
def shutdown():
    rospy.loginfo('stop the robot')
    if sendGoal is True:
        client.cancel_goal()
    rospy.sleep(2.0)

```

3. 导航

主函数

注册节点

注册节点关闭函数

注册导航动作客户端

定义导航目标

发送导航目标

等待导航完成

日志记录

```python
if __name__=='__main__':
    rospy.init_node("test_nav")
    rospy.on_shutdown(shutdown)
    client = actionlib.SimpleActionClient('move_base', \
        MoveBaseAction)
    sendGoal = False
    client.wait_for_server()
    while not rospy.is_shutdown():
        for pose in waypoints:
            rospy.loginfo('start')
            goal = goal_pose(pose)
            client.send_goal(goal)
            sendGoal=True
            isOnTime = client.wait_for_result(rospy.Duration(300))
            if not isOnTime:
                rospy.loginfo('TimeOut')
            state = client.get_state()
            if state == GoalStatus.SUCCEEDED:
                rospy.loginfo('well done')
            sendGoal=False
            rospy.sleep(5.0)

```