Unitree Nav Ros 工程说明

1.基础功能

uintree_nav_ros_node是一个导航节点，会根据地图和定位信息，规划机器人到给定目标点的全局和局部路径和速度，完成点到点的运动规划，其中目标点由Rviz中2D Nav Goal按键给出。它有以下基础功能：

1）规划机器人到目标点的局部路径

2）速度规划，发布目标速度话题"/untreated_cmd_vel"

3）Rviz可视化路径和机器人位姿

2.算法描述

2.1 状态机

根据机器人所处任务阶段不同，设置以下状态：

1）WAITING： 等待任务，静止不动

2）TURN_TARGET： 接到任务后转向目标点方向

3）WALKING： 正在朝目标点行走

4）ADJUST_ORIENTATION： 到达目标点后将机器人摇摆角对齐指定的目标点摇摆角

5）SLOW_DOWN： 到达目标点后减速到静止不动状态

状态改变当：

1）收到任务时，WAITING 变为 TURN_TARGET

2）机器人摇摆角转到目标点方向时，TURN_TARGET 变为 WALKING

3）机器人到达目标点时，WALKING 变为 SLOW_DOWN

4）机器人速度减为0时，SLOW_DOWN 变为 WAITING

2.2 路径规划

目标点在base系下的位置是机器人的行走方向，路径为一条直线。

其中，机器人到目标点的距离是欧式距离：

$$D_{robot-to-goal} = \sqrt{P_{x_{goal}}^2 + P_{y_{goal}}^2}$$

机器人与目标点的摇摆角是有方向的：

$$\Delta\theta_{robot-to-goal} = atan2(P_{y_{goal}}, P_{x_{goal}})$$

2.3 速度规划

计算机器人与目标点距离差值，做比例控制，角速度：

$$V_{linear} = P_{linear} * D_{robot-to-goal}$$

角速度：

$$W_{angular} = P_{angular} * \Delta\theta_{robot-to-goal}$$

3.代码说明

状态机 C++ 复制代码

```cpp
/** enum **/
enum Robot_index : const int // 机器人状态
{
    WAITTING = 1, // 正在等待任务
    TURN_TARGET, // 正在转向目标点
    WALKING, // 正在行走
    ADJUST_ORIENTATION, // 正在调整姿态
    SLOW_DOWN // 正在减速
};

// 状态改变
if (robot_index_ == TURN_TARGET) // 机器人处于首次转向状态
{   // 判断机器人是否首次转向完成
    if (fabs(robot_target_base_.tolerance_angle) < first_turn_target_tolerance_)
        robot_index_ = WALKING; // 机器人状态置为行走
}
else if (robot_index_ == WALKING) // 机器人处于行走状态
{   // 判断机器人是否到达目标点
    if (robot_target_base_.distance_robot_goal <= arrive_tolerance_)
    {
        robot_index_ = SLOW_DOWN; // 机器人状态置为减速
        if (isStop()) // 是否减速完成
            return true; // 返回到达目标点
    }
}
```

路径规划 C++ 复制代码

```cpp
// 转换坐标系
if (!transformPosePositioin("map", "base_link", robot_target_map_.pose, robot_target_base_.pose, listener_))
    return false;
// 计算目标点与机器人的方位角
robot_target_base_.tolerance_angle =
    atan2(robot_target_base_.pose.position.y, robot_target_base_.pose.position.x);
// 计算目标点与机器人的距离
robot_target_base_.distance_robot_goal =
    hypot(robot_target_base_.pose.position.x, robot_target_base_.pose.position.y);

```

速度规划 C++ 复制代码

```cpp
// 线速度规划
objective_vel.linear.x =
    (robot_target_base_.distance_robot_goal > 0.3) ? 0.3 : (1 *
fabs(robot_target_base_.distance_robot_goal));
// 角速度规划
objective_vel.angular.z =
    ((fabs(robot_target_base_.tolerance_angle) > 1) ? 1 : (1 *
fabs(robot_target_base_.tolerance_angle))) *
ComputeSign(robot_target_base_.tolerance_angle);
```

4.测试

4.1 测试说明

确认slam节点开启并正常运行、controller节点开启并正常运行、smooth_control_node节点开启并正常运行、LCM通讯通畅。

流程 C++ 复制代码

```
$ roscore
$ cd catkin_ws/
$ catkin_make
$ source devel/setup.sh
$ rosrun unitree_nav_ros unitree_nav_ros
```

4.2 测试结果

可以完成点到点的运动规划。

可以支持修改导航精度。

支持Rviz可视化。

目前不支持到达目标点后将机器人摇摆角对齐指定的目标点摇摆角。