**张 文泰**

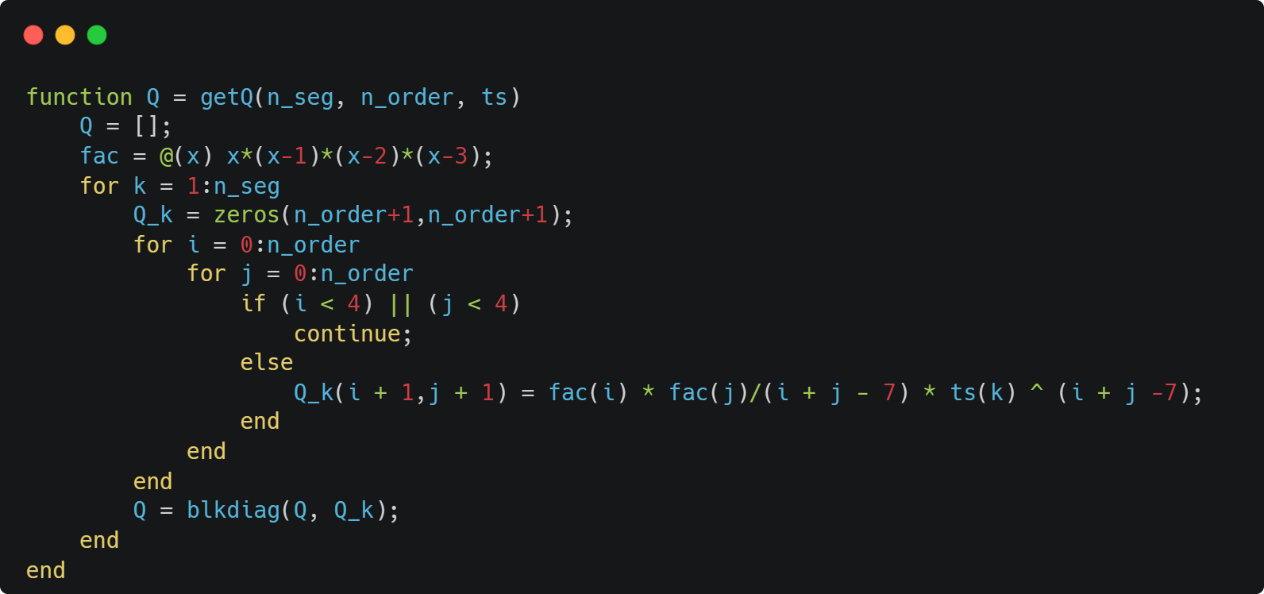
**学号：21009101463**

**机器人路径规划**

作业六 - 轨迹优化

MATLAB作业部分

1. 程序分析



getQ函数

函数getQ用于计算最小加速度规划的矩阵 Q，函数输入路径被分成的段数n\_seg、每个路径段上用于拟合轨迹的多项式的阶数n\_order、每个路径段的时间长度ts。

函数中通过fac函数定义了一个简单的阶乘表达式，通过循环为每个路径段计算 Q\_k 矩阵：对于每个路径段使用一个双重循环来填充 Q\_k 矩阵。因为只有多项式的阶数大于等于 4 时，才能保证拟合的多项式具有至少 4 次连续可导性，所以在这个循环中，只有当 i 和 j 都大于或等于 4 时，才会计算 Q\_k 的元素。 在计算完每个 Q\_k 矩阵后，使用 blkdiag 函数将它们放置在一个大的块对角矩阵 Q 中，并最终返回该矩阵。



getM函数

getM函数的输入参数与getQ函数相同，包括：路径被分成的段数n\_seg、每个路径段上用于拟合轨迹的多项式的阶数n\_order、每个路径段的时间长度ts。函数通过coeff矩阵定义了一个系数矩阵，用于计算 M\_k 矩阵的元素。然后，函数通过循环为每个路径段计算 M\_k 矩阵：使用一个双重循环来填充 M\_k 矩阵。在这个循环中，根据coeff矩阵的定义，M\_k 的对角线上的元素直接复制，而非对角线上的元素根据给定的时间间隔 t 进行计算。

与上一个函数相同，在计算完每个 M\_k 矩阵后，使用blkdiag函数将它们放置在一个大的块对角矩阵 M 中，并最终返回该矩阵。



getCT函数

函数 getCt用来生成约束矩阵 Ct，即用于描述轨迹端点（起点和终点）和轨迹段之间连接约束的矩阵。这个函数的输入参数包括路径被分成的段数n\_seg、每个路径段上用于拟合轨迹的多项式的阶数n\_order。函数中初始化 Ct为一个零矩阵，其行数和列数根据给定的路径段数和阶数来确定。然后，函数通过循环为 Ct矩阵的每个元素赋值。

1. 起始约束（Start Constraint）：前4行设置为单位矩阵，以确保起始点的位置和前三阶导数被固定。
2. 中间约束（Intermediate Constraints）：循环中的第二部分用于处理路径段之间的约束。它通过在每个路径段内部设置固定和自由变量来连接不同路径段之间的约束。
3. 终止约束（Terminal Constraint）：最后4行设置为单位矩阵，以确保终点的位置和前三阶导数被固定。

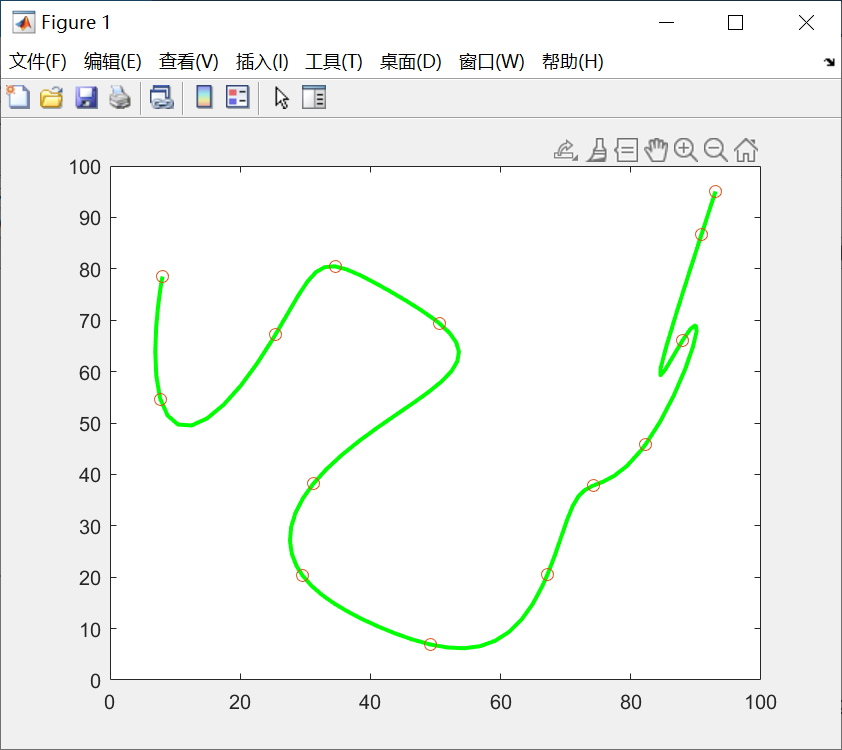


getAbeq函数

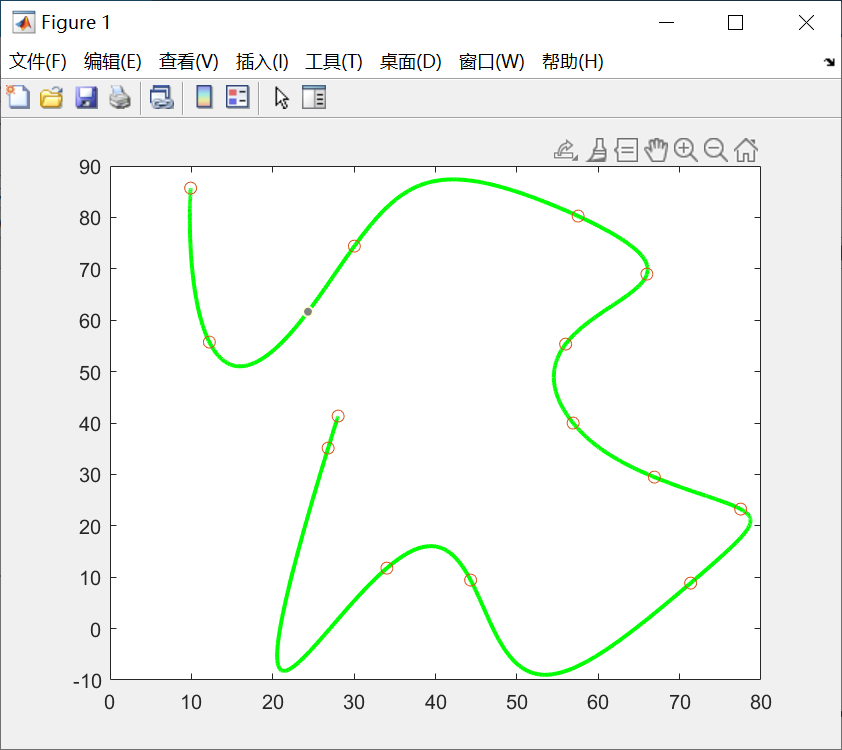
函数getAbeq用于生成等式约束矩阵 Aeq 和等式约束向量 beq以描述最小加速度规划问题中的起始、终止条件、中间路点、以及路径段之间的连续性约束。函数主要步骤包括：

1. 起始条件约束：通过调用 getCoeff 函数得到起始条件所对应的多项式系数，并将其放置在 Aeq 和 beq 中。
2. 终止条件约束：同样地，通过调用 getCoeff 函数得到终止条件所对应的多项式系数，并将其放置在 Aeq 和 beq 中。
3. 中间路点约束：将中间路点的位置信息放置在 Aeq 和 beq 中。
4. 路径段连续性约束：对于每两个相邻的路径段，设置位置、速度、加速度和 jerk 的连续性约束。
5. 组合所有约束：将所有约束组合成 Aeq 和 beq。
6. 实验结果

运行hw1\_1.m与hw1\_2.m，使用鼠标左键在图中选点，随机选取几个点后按回车，会出现规划好的路径，如图所示。



hw1\_1.m运行结果



hw1\_2.m运行结果

ROS作业部分

1. 程序分析



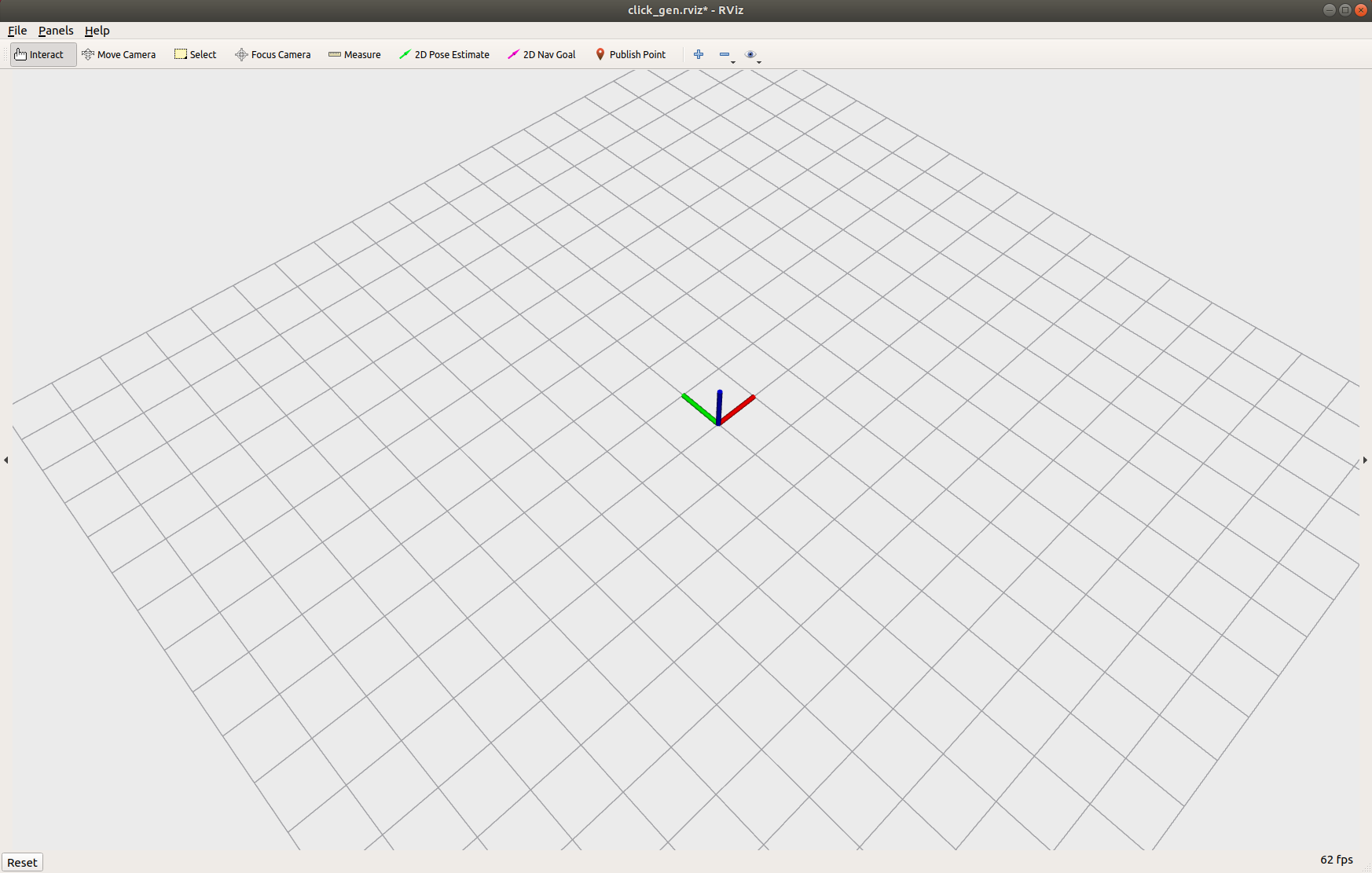
minimumJerkTrajGen函数

这段代码的主要思路是利用多项式的最小加速度（minimum jerk）轨迹生成算法来计算给定初始和终点状态下的光滑轨迹。代码思路如下：

1. 初始化矩阵和向量：首先，初始化了两个矩阵 和 ，它们分别用于构建系数矩阵和常数矩阵。
2. 处理初始状态：将初始位置、速度和加速度放入常数矩阵 的前三行。
3. 循环处理每个时间段：对于每个时间段，生成了一个的矩阵 和一个的矩阵 ，它们分别代表了当前时间段内位置、速度和加速度的关系，然后将这些矩阵填入系数矩阵 的相应位置。
4. 处理终点状态：将终点的位置、速度和加速度放入常数矩阵 的最后三行。
5. 求解系数矩阵：通过求解线性方程组 ，得到了系数矩阵 。

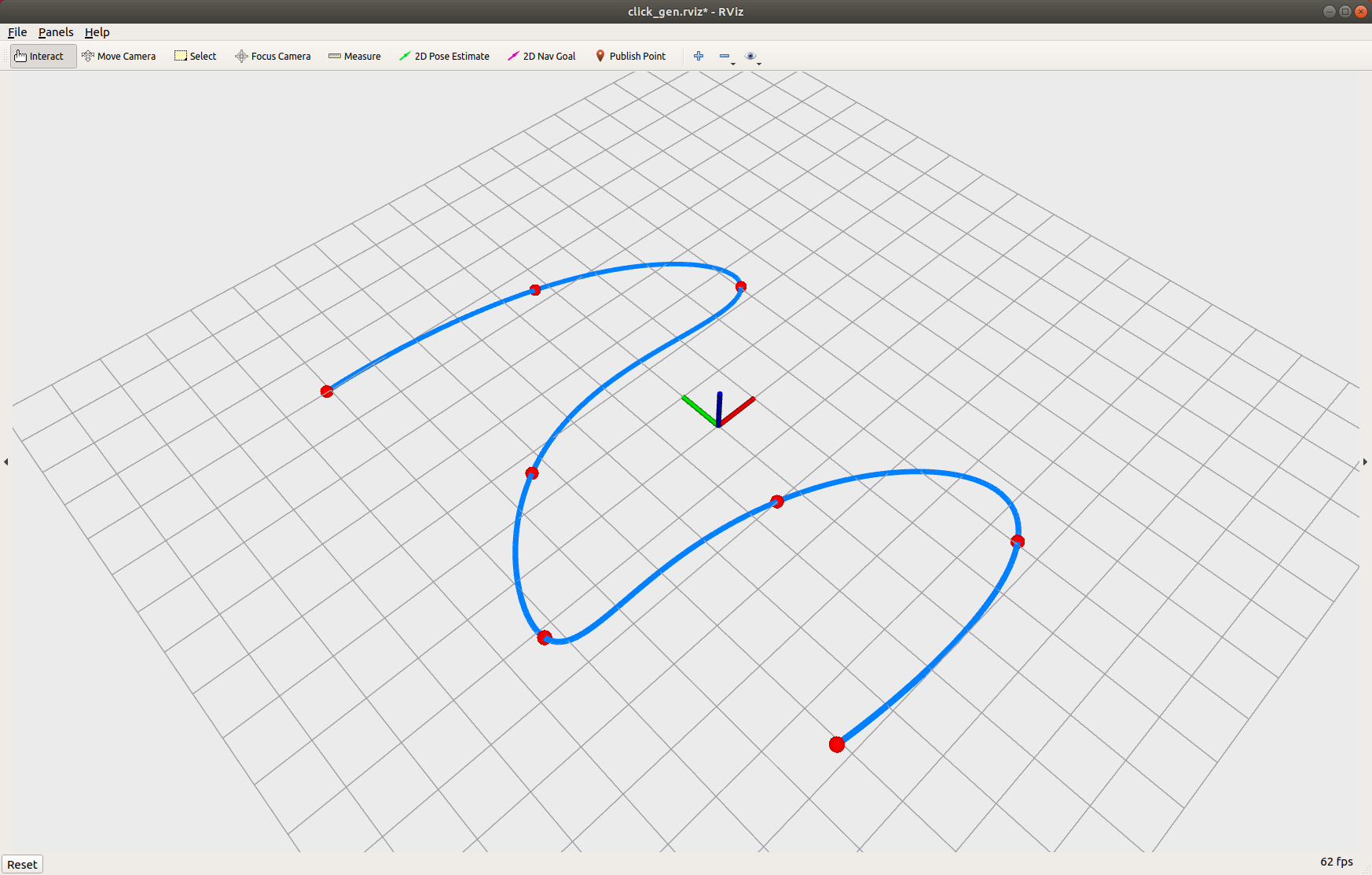
简单来讲就是，这段代码通过循环处理每个时间段，构建了一个线性方程组，然后通过求解该方程组得到了最小加速度轨迹的系数矩阵。

1. 实验结果



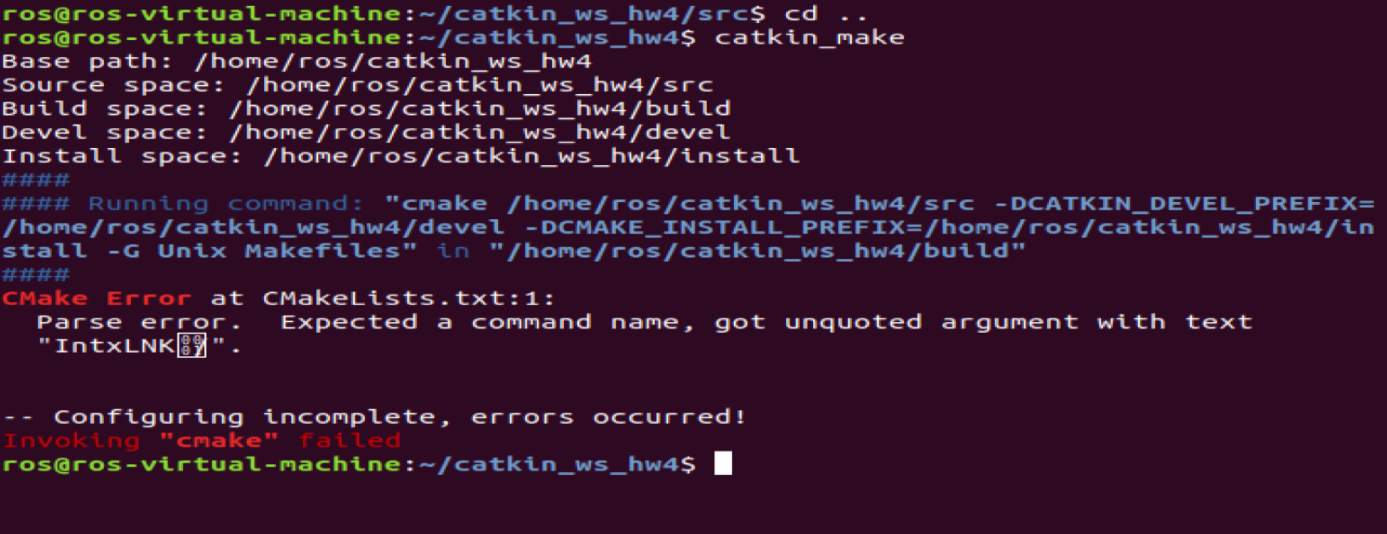
RViz运行成功

如图所示，程序成功运行，地图显示在RViz中。接下来使用2D nav goal工具在图中选取出发点并依次选取路径点，可以看到在图中显示出了平滑的轨迹，运行结果如图所示：



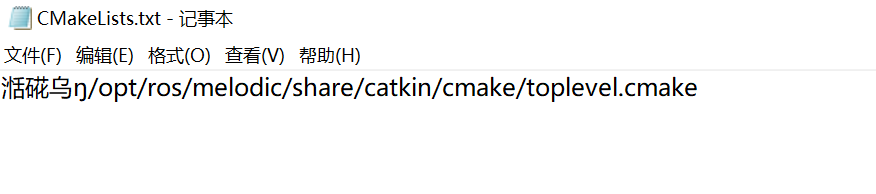
轨迹运行结果

1. 实验中遇到的问题及解决方案



报错现象

如图所示，我在进行工作空间编译时遇到了报错，根据报错内容可以得知是CMakeLists.txt存在问题，于是我去查看该文件内容，如图所示：



CMakeLists.txt内容

可以发现该文件出现编码问题，开头并非使用UTF-8编码导致Ubuntu无法识别，在修正之后依然出现报错情况，于是我新建工作空间，在将作业包中的内容导入Ubuntu工作空间时，没有导入CMakeLists.txt，而是使用工作空间初始化后自动生成的CMakeLists.txt，编译之后程序成功运行，问题得到解决。