**移动机器人规划与控制期末报告**

学号：22354073 姓名：梁凯 提交日期：2025/1/18

**提交的Github仓库链接：**

<https://github.com/bnvnbnvn/Mobile-Robot-Planning-Assignment-and-control>

**实验一：**

图形用户界面, 文本, 应用程序

描述已自动生成

在四旋翼飞行器的动力学模型中，v\_dot 和 omega\_dot 分别表示线速度和角速度的导数（即加速度和角加速度）。

**1.1.1** **v\_dot代码原理：**

根据牛顿第二定律：

式中, 与右上角的代表这是地面坐标系下的向量，右上角的代表这是机体坐标系下的向量。由于拉力是由螺旋桨产生的，与四旋翼飞行器固连，故在机体坐标系下表示。

等式两边同时除以：

将机体坐标系下的拉力转换到地面坐标系，左乘旋转矩阵即可：

**1.1.2** **v\_dot代码变量解释：**

**R \* thrust \* Eigen::Vector3d::UnitZ()：**

推力向量在飞行器机体坐标系下的方向是 R.col(2)（即 z 轴方向），乘以总推力得到推力向量。R 是飞行器的姿态矩阵，将推力从机体坐标系转换到世界坐标系。thrust 是总推力，Eigen::Vector3d::UnitZ() 是单位向量 [0, 0, 1]，表示推力方向。

**g\_ \* mass\_ \* Eigen::Vector3d::UnitZ()：**

重力作用在地面坐标系下的方向是 -z 轴方向，乘以质量和重力加速度得到重力向量。g\_ 是重力加速度，mass\_ 是飞行器的质量，Eigen::Vector3d::UnitZ() 是单位向量 [0, 0, 1]，表示重力方向。

**external\_force\_：**

总外力，即外部环境对飞行器施加的力。

**resistance \* vnorm：**

resistance 是阻力系数，vnorm 是速度的单位向量，二者的积为空气阻力。

**1.2.1** **omega\_dot代码原理：**

绕质心的转动由欧拉方程描述，即：

式中, 为刚体所受的合外力矩；为3×3的惯性矩阵；为刚体的角速度。其表达的含义为：作用在刚体上的合力矩 使得刚体以角速度、角加速度 旋转。

为求解将等式作如下变形：

**1.2.2** **omega\_dot代码变量解释：**

**moments：**

四个旋翼产生的总力矩。

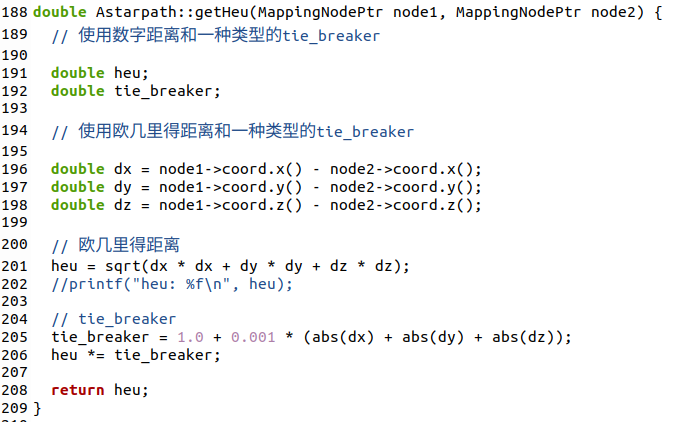
**omega\_vee \* J\_ \* cur\_state.omega：**

omega\_vee 是角速度的反对称矩阵，J\_ 是惯性矩阵，cur\_state.omega 是当前角速度。三者相乘可得到科里奥利力。

**external\_moment\_：**

外部环境对飞行器施加的力矩。

**实验二：**



这段代码实现了 A\* 算法中的启发式函数 getHeu()，用于估计从当前节点到目标节点的代价。启发式函数在 A\* 算法中的作用是影响路径搜索的效率和效果。

启发式函数 getHeu() 的具体作用是估计从当前节点 node1 到目标节点 node2 的代价。常用的启发式函数包括欧几里得距离、曼哈顿距离等。这里使用的是欧几里得距离，并加入了 tie\_breaker 来打破平局。

首先根据欧几里得距离公式计算两点之间的直线距离：

作为启发式函数的估计值。

然后加入tie\_breaker，其作用是防止 A\* 算法在代价相同的情况下出现路径选择上的平局。tie\_breaker 的值略大于 1，增加了一个小的偏移量，使得启发式估计值略微偏大，有助于算法更快地收敛到最优路径。此处 tie\_breaker 是通过 1.0 + 0.001 \* (abs(dx) + abs(dy) + abs(dz)) 计算的，增加了一个与坐标差值成正比的小偏移量。

文本

描述已自动生成

此段代码实现了 A\* 算法，用于在搜索空间中找到从起点到终点的最优路径。对这段代码的解释如下：

A\* 算法是一种启发式搜索算法，通过结合实际代价和启发式估计来找到从起点到终点的最优路径。主循环的作用是不断从OpenSet中取出代价最小的节点，扩展其邻居节点，并更新邻居节点的信息，直到找到终点或OpenSet为空。

**1.弹出 g+h 最小的节点：**

Openset是一个 multimap，按 f\_score = g\_score + h\_score升序排序。取currentPtr = currentPair->second，即Openset中f\_score最小的节点，再从 Openset 中弹出它。这么写的作用是保证每次扩展的都是当前代价最小的节点，确保 A\* 算法的最优性。

**2.判断是否是终点：**

如果当前节点是终点节点，则设置 terminatePtr 为当前节点，并返回true，表示找到路径。 这么写的作用是提前终止搜索，节省计算资源。

**3.****拓展当前节点：**

调用AstarGetSucc函数，获取当前节点的所有邻居节点和对应的边代价，用以拓展当前节点。neighborPtrSets存储邻居节点，edgeCostSets存储边代价。

**4.填写信息，完成更新：**

如果邻居节点的id = 0，表示该节点未被访问过，那么：

①更新邻居节点的g\_score和f\_score；

②设置邻居节点的父节点为当前节点；

③将邻居节点插入Openset；

④将邻居节点的id设置为1，表示已在Openset中。

这么写的作用是：更新邻居节点的信息，确保每个节点的 `g\_score` 和 `f\_score` 是最优的；使用id标记节点的状态，避免重复访问。

注意到if (id = 0) 和else if (id = 1) 之后的代码块有很大相似性，所以如果是不单单为了补充此处，可以更换判断条件将代码重写。换言之，此处的逻辑也可以表示为：

文本

描述已自动生成

**实验三：**

文本

描述已自动生成

均方误差为：0.0331877009398189

运行时间为：74.3193275928497

轨迹长度为：32.9311702747195

是否发生了碰撞：0

综合评分：0.191564051768357

**实验四：**

（有则说明一下，没有则写无）

无