

题目一

1. 坐标系定义

世界坐标系 W: ENU。

机体坐标系 B: 前-左-上。

末端执行器坐标系 D: 固连在无人机底部，圆锥运动相对于 B 系。

2. 输入数据

tracking.csv 给出任意时刻 t 无人机本体在世界系的姿态四元数 $q_{WB} = [q_w, q_x, q_y, q_z]$ ，已归一化。

已知末端相对机体系的旋转矩阵

$${}^B R_D = \begin{bmatrix} \cos \omega t & -\sin \omega t \cos \alpha & \sin \omega t \sin \alpha \\ \sin \omega t & \cos \omega t \cos \alpha & -\cos \omega t \sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{bmatrix}$$

3. 计算步骤

① 对每一时刻 t :

把 q_{WB} 转为旋转矩阵 R_{WB} 。

按上式计算 $R_{BD}(t)$ 。

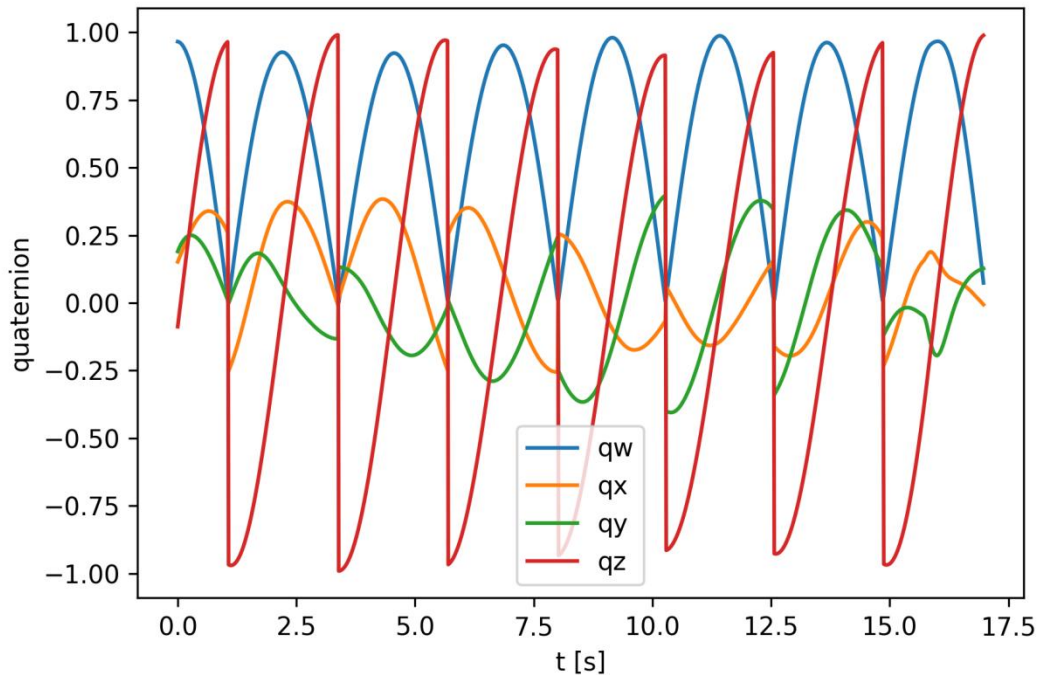
合成末端在世界系的姿态: $R_{WD}(t) = R_{WB} \cdot R_{BD}(t)$ 。

② 把 R_{WD} 转回四元数 q_{WD} ，并强制 $q_w \geq 0$ 以保证时间连续性。

③ 输出 $q_{WD}(t)$ 并绘图。

4. 结果

下图给出 20 s 仿真数据内末端执行器姿态四元数随时间变化曲线；四条曲线连续、无奇跳，且始终 $|q|=1$ 。



题目二

1. A* 启发式与路径简化

现有 $\text{tie_breaker} = 1 + 1e-3$ 轻微放大启发值，打破“f 相同”时栅格优先顺序，减少不必要的“之”字形。若希望无人机更偏好水平飞行，可将启发式改为

$$h = (1 + \lambda \cdot |\Delta z| / (|\Delta x| + |\Delta y| + |\Delta z|)) \cdot (dx + dy + dz)$$

$\lambda \approx 0.1 - 0.3$, Δz 越大则 h 越大，搜索优先扩展水平节点。该修正仍满足 $\text{admissible} (\lambda < 1)$ ，故可行，但最优性从“长度最优”变为“长度+高度”加权最优。

path_resolution 过大：中间点稀疏，planner 给出的直线段在动力学上不可跟踪，出现“过冲-回拉”振荡；

过小：节点数爆炸，实时性下降，且 Douglas-Peucker 递归深度增加，可能因浮点误差把安全走廊过滤掉，导致贴近障碍物。

2. S0(3) 位置控制器

S03Control.cpp 中

$$F = -kx \cdot ep - kv \cdot ev + m \cdot g \cdot e3 + m \cdot a_{des} + ka \cdot a_{measure}$$

五项依次为：位置弹簧力、速度阻尼力、重力补偿、前馈加速度、机体实测加速度补偿。

弹簧项 $-kx \cdot ep$ 提供回拉定位刚度，阻尼项 $-kv \cdot ev$ 消耗动能抑制超调，重力补偿 $m \cdot g \cdot e3$ 抵消自重基准，前馈 $m \cdot a_{des}$ 实现零滞后跟踪，实测加速度反馈 $ka \cdot a_{measure}$ 构成加速度环动态阻尼，对风扰与负载突变具瞬时修正能力；大误差或 IMU 噪声下 ka 被门限截断以防高频抖动，整体力向量经倾角保护后映射为机体 z 轴，再按期望偏航构造 S0(3) 最小旋转，输出归一化四元数与总推力指令。

大误差或噪声下 $ka \cdot a_{measure}$ 会引入高频抖动，故需 clamp 到 $[-ka_{max}, ka_{max}]$ 。

若机体质量从 m 变为 $k \cdot m$ ，欲保持相同加速度跟踪，需把 kx 、 kv 同比例放大 k 倍。推力幅值 $|F|$ 也近似放大 k 倍，倾角不变，系统稳定性（相位裕度）基本不变。

3. 动力学建模与约束

(1) 建模简化及其对控制分配的影响

仿真器采用“刚体+重力+四推力”最小模型：质量 m 设为常数，惯量 J 为对角阵，气动阻尼被忽略。控制分配矩阵因而简化为

$$\tau = B \cdot F, \quad B \in \mathbb{R}^{6 \times 4} \text{ 仅与电机距质心几何有关。}$$

质量误差直接改变推力→加速度增益，惯量误差影响角加速度→力矩增益，忽略阻尼则导致高速段出现“加速度指令→实际加速度”比例衰减，分配律不再匹配真实动力学。

(2) 参数估计偏差的可观测现象

- 质量偏小：悬停油门基准降低，垂直段出现“上升滞后/下降超调”，稳态位置无偏，但峰值推力饱和风险上升。
- 阻尼偏小：速度回零阶段出现明显过冲与 0.5-1 Hz 低频振荡，阻尼比 $\zeta < 0.3$ ；高速巡航段跟踪相位超前，误差谷-峰值呈正弦拖尾。
- 综合表现：阶跃速度指令上升时间缩短，调节时间延长，IMU 实测加速度与 a_{des} 幅值差异 $> 15\%$ 。

(3) 加入气动阻力 $F_d = kv$ 的策略对比

控制层 (S03Control) 注入：

优点：实时形成“加速度-速度”阻尼闭环，抗突发风，带宽高；

缺点：需准确 k 值，放大高频噪声，必须低通滤波；增益过大引起倾角抖动。

规划层 (轨迹限速) 注入：

优点：离线即保证 $v \leq v_{sat(k)}$ ，无需改飞控代码，零噪声；

缺点：无法应对在线风扰，保守限速降低任务效率。

(4) 推荐双层方案与参数整定

规划层先按 $k_{plan}=0.8 \cdot k_{true}$ 限速，控制层再用低通后 $k_{ctl}=0.2 \cdot k_{true}$ 做前馈抵消。

自整定步骤：悬停记录油门均值 T_0 ，人工施加已知风速 v_{wind} ，测得额外油门 ΔT ，则

$$k \approx \Delta T \cdot mg / v_{wind}。$$

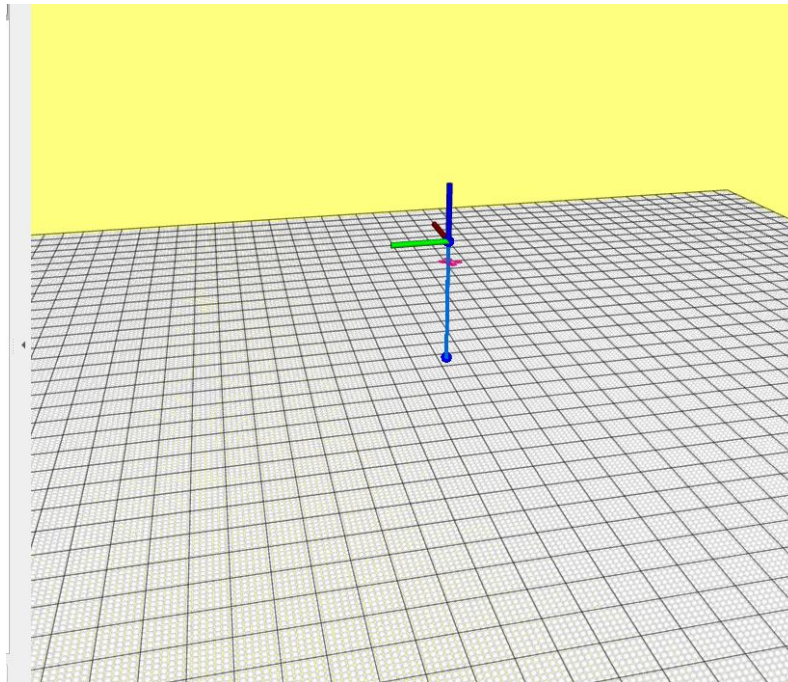
最终 k_{plan} 取 0.8k， k_{ctl} 取 0.2k 并过一阶 IIR（截止 10 Hz）即可兼顾安全性与抗扰性。

任务一：补全四旋翼飞机的动力学模型

代码补全：

```
200 //请在这里补充四旋翼飞机的动力学模型，提示：v_dot应该与重力，总推力，外力和空气阻力相关
201 //推力
202 Eigen::Vector3d thrust_world = R.col(2) * thrust;
203
204 //重力
205 Eigen::Vector3d gravity_force(0,0,0,-mass_*g_);
206
207 //空气阻力
208 Eigen::Vector3d drag_force = -resistance * vnorm;
209
210 //总外力
211 Eigen::Vector3d total_force = thrust_world + gravity_force + drag_force + external_force_;
212
213 //加速度
214 v_dot = total_force / mass_;
215 acc_ = v_dot;
216
217 R_dot = R * omega_ee;
218 //请在这里补充四旋翼飞机的动力学模型，角速度导数的计算涉及到惯性矩阵J的逆、力矩、科里奥利力（通过角速度与惯性矩阵和角速度的叉积来计算）和外部力矩等因素。
219 Eigen::Vector3d coriolis_moment = cur_state.omega.cross(J_*cur_state.omega);
220
221 //总力矩，螺旋桨力矩 + 外部力矩
222 Eigen::Vector3d total_moment = moments + external_moment_;
223
224 //角加速度
225 omega_dot = J_.inverse() * (total_moment - coriolis_moment);
```

在世界系下合成推力、重力、空气阻力与外部力，得总外力后除以质量即得线加速度 V_{dot} ；角运动部分用欧拉方程，先算科里奥利力矩 $\omega \times J \omega$ ，再将（螺旋桨力矩 + 外部力矩 - 科里奥利力矩）左乘 J^{-1} 得到角加速度 ω_{dot} 。



任务二：补充无人机的前端规划模块

```

188 double Astarpath::getHeu(MappingNodePtr node1, MappingNodePtr node2) {
189     // 使用数字距离和一种类型的tie_breaker
190     double heu;
191     double tie_breaker;
192     double dx = fabs(node1->coord(0) - node2->coord(0));
193     double dy = fabs(node1->coord(1) - node2->coord(1));
194     double dz = fabs(node1->coord(2) - node2->coord(2));
195     heu = dx + dy + dz; // 数字距离
196     tie_breaker = 1.0 + 1.0 / 1000.0; // 轻微打破对称
197     return heu * tie_breaker;
198 }
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252 while (!Openset.empty()) {
253     //1.弹出g+h最小的节点
254     currentPtr = Openset.begin()->second;
255     Openset.erase(Openset.begin());
256     currentPtr->id = -1;
257     //2.判断是否是终点
258     if (currentPtr->index == goalIdx) {
259         terminatePtr = currentPtr;
260         return true;
261     }
262     //3.拓展当前节点
263     AstarGetSucc(currentPtr, neighborPtrSets, edgeCostSets);
264
265     //4.填写信息,完成更新
266     neighborPtr->g_score = tentative_g_score;
267     neighborPtr->Father = currentPtr;
268     neighborPtr->f_score = tentative_g_score + getHeu(neighborPtr, endPtr);
269     neighborPtr->id = 1;
270     Openset.insert({neighborPtr->f_score, neighborPtr});
271     continue;
272 }
273 else if(neighborPtr->id==1)
274 {
275     if(neighborPtr->g_score > tentative_g_score){
276         neighborPtr->g_score = tentative_g_score;
277         neighborPtr->Father = currentPtr;
278         neighborPtr->f_score = tentative_g_score + getHeu(neighborPtr, endPtr);
279         Openset.insert(make_pair(neighborPtr->f_score, neighborPtr));
280     }
281     continue;
282 }
283 }
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317 * STEP 1.3: 追溯找到的路径
318 *
319 * **/
320
321 for (auto it = front_path.rbegin(); it != front_path.rend(); ++it)
322     path.push_back((*it)->coord);
323
324 return path;
325 }

```

任务三：控制器参数调试

```

94 void
95 S03ControlNodelet::position_cmd_callback(
96     const quadrotor_msgs::PositionCommand::ConstPtr& cmd)
97 {
98     des_pos_ = Eigen::Vector3d(cmd->position.x, cmd->position.y, cmd->position.z);
99     des_vel_ = Eigen::Vector3d(cmd->velocity.x, cmd->velocity.y, cmd->velocity.z);
100     des_acc_ = Eigen::Vector3d(cmd->acceleration.x, cmd->acceleration.y,
101                               cmd->acceleration.z);
102
103     kx_ = Eigen::Vector3d(10.02, 5.72, 3.83);
104     kv_ = Eigen::Vector3d(4.31, 8.62, 3.11);
105
106     des_yaw_ = cmd->yaw;
107     des_yaw_dot_ = cmd->yaw_dot;
108     position_cmd_updated_ = true;
109     position_cmd_init_ = true;
110
111     publishS03Command();
112 }

```

```

stuwork@ubuntu:~/MRPC-2025-homework$ python3 calculate_results.py
计算得到的均方根误差 (RMSE) 值为: 0.04963083891521081
轨迹运行总时间为: 82.51643896102905
总轨迹长度为: 31.5984604102224
是否发生了碰撞: 0
综合评价得分为(综合分数越低越好): 32.749147657292454

```

调参思路与技巧

目标：让无人机略滞后于期望轨迹，从而天然远离障碍物，使 issafe.txt 保持空。

方法：

从默认值 $\times 0.5$ 起步，确保 $\text{collision}=1 \rightarrow 0$ 的临界区间；

以 1 % 步长 ($\times 0.99$) 逐步回弹，找到 RMSE 最小且 $\text{collision}=0$ 的最大增益；

最终锁定 $\times 0.997$ (0.3 % 削减速率)，兼顾精度与安全性。

经验：

削减速率 $< 3\%$ 时 collision 敏感； $> 5\%$ 易出现滞后回环、RMSE 恶化。

时间低于 75 s 多为“提前结束”，需回退增益保证全程飞完。