



智能控制技术 实验报告

实验名称	专家控制
实验地点	教 4 302
姓 名	董佳泽
学 号	3230104322
实验日期	November 27, 2025
指导老师	刘山

Contents

1 实验目的与要求	1
1.1 实验背景与意义	1
1.2 实验任务	1
2 系统建模与问题分析	3
2.1 USV 非线性动力学模型	3
2.1.1 模型参数物理意义	3
2.1.2 状态空间方程推导	3
2.1.3 S-Function 模型	4
2.2 环境干扰模型	5
2.3 性能评价指标定义	6
3 控制器设计与实现	7
3.1 普通 PID 控制器设计	7
3.1.1 控制器结构与思路	7
3.1.2 Simulink 实现	7
3.1.3 仿真结果	9
3.2 直接专家控制器设计	10
3.2.1 规则库设计	10
3.2.2 推理机设计	12
3.2.3 仿真结果	13
3.3 专家 PID 控制器设计	14
3.3.1 规则库设计	14
3.3.2 推理机设计	16
3.3.3 仿真结果	16
4 性能分析	17
4.1 标准工况下的对比分析	17
4.2 鲁棒性分析：参数摄动	18
4.3 抗干扰性分析：环境噪声增强	19
4.4 跟踪性能分析：正弦信号跟踪	20
5 结论	21

6 心得体会	22
A 附录 1: 附件结构	23
B 附录 2: 推理机代码	23

1 实验目的与要求

1.1 实验背景与意义

随着海洋智能装备技术的飞速发展，无人水面艇 (Unmanned Surface Vehicle, USV) 在海洋环境监测、水文测绘及海防巡逻等领域展现出广阔的应用前景。航向控制作为 USV 实现自主循迹与作业的最底层核心功能，其性能直接决定了上层任务的成败。

然而，USV 的航向控制面临着严峻的挑战，主要体现在以下三个方面：

- 强非线性与大惯性：**根据本实验给定的动力学模型 $T\dot{\psi} + KH(\dot{\psi}) = K\delta + d(t)$ ，船舶具有显著的非线性阻尼特性 $H(\dot{\psi}) = \alpha\dot{\psi} + \beta\dot{\psi}^3$ ，且追随性指数 $T = 2.0s$ 意味着系统存在较大的惯性延迟。
- 复杂的环境干扰：**实际海况中存在风、浪、流等随机干扰。本实验中模拟了复合干扰 $d(t) = A_d \sin(\omega_d t) + n(t)$ ，其中包含持续的正弦波浪推力和高频白噪声，这对控制器的抗干扰能力提出了极高要求。
- 执行机构物理约束：**实际舵机存在物理幅值限制，本实验要求舵角 $|\delta| \leq 0.5 \text{ rad}$ 。传统的线性控制容易导致积分饱和现象，引发系统超调甚至失稳。

鉴于传统 PID 控制器在处理非线性约束和参数摄动时的局限性，本实验旨在引入专家控制策略。通过模拟人类驾驶员的智慧，采用“知识表示与推理运用分离”的架构，设计能够适应复杂工况的智能控制器，以提升 USV 系统的鲁棒性与自适应能力。

1.2 实验任务

本实验在 MATLAB/Simulink 仿真平台上，以 USV 非线性模型为研究对象。首先，设计**普通 PID 控制器**作为性能比较的基准；随后，根据误差 $e = \psi_{\text{ref}} - \psi$ 及其变化率 $\dot{e} = \dot{e}$ ，分别设计**直接专家控制器**与**专家 PID 控制器**；最终，从多个维度对上述控制策略进行综合性能对比。

1. 普通 PID 控制器设计（基准）

- 目标：**构建基于误差反馈的线性 PID 闭环控制系统。
- 要求：**作为性能对比的基准，在满足舵角限幅 $|\delta| \leq 0.5$ 的前提下，通过工程整定法确定一组标称参数，力求在标准工况下达到较好的跟踪效果。

2. 直接专家控制器设计

- 目标：**设计一个基于规则库直接输出舵角 δ 的智能控制器。

- **架构要求：**严格遵循“规则库与推理机分离”的原则。规则库需涵盖五种控制逻辑：
 - **饱和控制：**偏差很大时，打满舵，以求最快速度纠正。
 - **增强控制：**偏差中等且仍在变大，用中等舵角抑制其增长。
 - **保守控制：**偏差中等但已在缩小，用小舵角柔和纠正，防止超调。
 - **精细控制：**接近目标，启用类 PD 的精细控制，平滑稳定。
 - **微调控制：**在极小误差区，使用类 PID 控制消除稳态误差。
(虽然作业提示可施加微小恒力，但实验表明在存在扰动的情况下，保持类 PID 控制更能有效维持跟踪精度。)

3. 专家 PID 控制器设计

- **目标：**设计基于专家规则在线动态调整 PID 参数 (K_p, K_i, K_d) 的自适应控制器。
- **要求：**实现以下四种智能整定策略：
 - **抗饱和：**偏差极大，切换为 Bang-Bang 控制，绕过 PID。
 - **加速响应：**误差在扩大，增强 P 和 D 以抑制偏差增长，削弱 I 防积分饱和。
 - **抑制超调：**误差在缩小，减弱控制作用，使系统平稳接近目标。
 - **消除静差：**接近目标，增强积分作用以消除稳态误差，减弱 P 和 D 防振荡。

4. 系统仿真与多维性能对比

- **目标：**在标称工况及**参数摄动工况**（如 T 增大、 K 减小、噪声增强）下，从以下五个维度定量评估三种控制器的性能。
- **性能指标：**
 - **超调量：**反映系统的动态稳定性。
 - **调节时间：**反映系统的响应速度。
 - **稳态误差：**反映系统的控制精度。
 - **控制输入平滑性：**量化舵角的抖动程度，评估对执行机构的保护能力。
 - **鲁棒性：**评估控制器对模型参数变化和环境干扰的适应能力。

2 系统建模与问题分析

2.1 USV 非线性动力学模型

本实验的研究对象为一艘无人水面艇（USV），该系统包含显著的非线性阻尼项和外部环境干扰，其微分方程描述如下：

$$T\dot{r} + KH(r) = K\delta + d(t)$$

其中， ψ 为航向角， $r = \dot{\psi}$ 为转艏角速度， δ 为控制输入舵角。

2.1.1 模型参数物理意义

- T ：反映船舶的惯性特征。 T 值越大，船舶改变运动状态所需的时间越长，系统的响应速度越慢。
- K ：反映船舶的转向灵敏度。 K 值越大，相同的舵角产生的转向力矩越大。
- $H(r)$ ：描述了船舶在不同转速下的水动力阻尼特性。本模型采用三阶非线性描述：

$$H(r) = \alpha r + \beta r^3$$

其中 α 为线性阻尼系数， β 为非线性阻尼系数。非线性项的存在使得系统在大机动时的动态特性与线性模型存在显著差异。

2.1.2 状态空间方程推导

为了便于在 Simulink 中通过 S-Function 模块进行数值仿真及控制器设计，需将上述高阶微分方程转化为一阶状态空间形式。选取状态变量向量 $\mathbf{x} = [x_1, x_2]^T$ ，其中：

- $x_1 = \psi$ （航向角）
- $x_2 = r = \dot{\psi}$ （转艏角速度）

根据微分方程可以推导出系统的状态空间方程组：

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = \frac{1}{T} [K\delta - K(\alpha x_2 + \beta x_2^3) + d(t)] \end{cases}$$

该方程组即为系统仿真模型的核心数学描述。在本实验中，标称参数设定为： $T = 2.0s$, $K = 0.8s^{-1}$, $\alpha = 1.0$, $\beta = 0.5$ 。

2.1.3 S-Function 模型

根据上面推导出的状态空间方程，可以编写出 S-Function 的核心部分如下。

Listing 1: USV S-Function

```
1 function sys = mdlDerivatives(t,x,u)
2
3 % --- 获取模型参数 ---
4 T = 2; % 追随性指数
5 K = 0.8; % 旋回性指数
6 alpha = 1.0; % 非线性参数 alpha
7 beta = 0.5; % 非线性参数 beta
8
9 % --- 获取状态变量 ---
10 %  $x(1) = \psi$  (航向角)
11 %  $x(2) = r$  (转艏角速度)
12 r = x(2);
13
14 % --- 获取输入 ---
15 delta = u(1); % 控制输入: 舵角
16 dist = u(2); % 外部扰动:  $d(t)$ 
17
18 % --- 非线性函数  $H(r\_dot)$  ---
19 % 题目中  $H(\psi\_dot) = \alpha \psi\_dot + \beta \psi\_dot^3$ 
20 % 即  $H(r) = \alpha r + \beta r^3$ 
21 H_r = alpha * r + beta * r^3; %
22
23 % --- 状态方程推导 ---
24 % 1.  $\psi\_dot = r$ 
25 % 2.  $T * r\_dot + K * H(r) = K * delta + d(t)$ 
26 %  $\Rightarrow r\_dot = (1/T) * (K * delta + d(t) - K * H(r))$ 
27
28 dx1 = r;
29 dx2 = (1/T) * (K * delta + dist - K * H_r);
30
31 sys = [dx1; dx2];
```

2.2 环境干扰模型

实际海况中的干扰具有成分复杂、频带覆盖宽的特点。本实验采用复合干扰模型 $d(t)$ 来模拟风、浪、流对 USV 航向的影响：

$$d(t) = A_d \sin(\omega_d t) + n(t)$$

该干扰模型由两部分组成，对控制系统提出了不同的挑战：

1. **低频周期性干扰** ($A_d \sin(\omega_d t)$)：模拟海浪的周期性推力。

挑战：它会导致系统输出产生持续的强迫振荡。若控制器缺乏足够的积分作用或自适应能力，将产生难以消除的动态稳态误差。

2. **高频随机噪声** ($n(t)$)：模拟传感器测量噪声或随机风干扰，表现为均值为 0、方差为 0.01 的白噪声。

挑战：它会被控制算法中的微分项 (D) 显著放大，导致控制输入产生高频颤振，从而加剧执行机构的机械磨损。

在 simulink 中，白噪声部分通过 **Band-Limited White Noise** 模块实现，该模块参数设置如图1所示。



Figure 1: 白噪声模块参数设置

查阅 Matlab 帮助文档可知，噪声的方差满足以下公式：

$$D = \frac{P}{T}$$

其中 D 表示噪声方差； P 表示噪声功率； T 表示采样周期。因此在 $T = 0.1$ 的前提下，为了保证 $D = 0.01$ ，此处设置 $P = 0.001$ 。

2.3 性能评价指标定义

为了定量评估不同控制策略的优劣，本实验设计了 Monitor 模块和评估函数，对以下关键指标进行计算：

1. **超调量**：描述系统的瞬态响应稳定性。

$$M_p = \frac{\max(\psi(t)) - \psi_{ref}}{\psi_{ref}} \times 100\%$$

其中 ψ_{ref} 为目标设定值。

2. **调节时间**：描述系统的响应速度。定义为输出 $\psi(t)$ 进入并保持在目标值 $\pm 5\%$ 误差带内的最短时间。
3. **稳态误差**：由于存在正弦干扰，系统稳态响应并非恒定值。因此采用末端均值法计算，取仿真结束前 20% 时间段内误差绝对值的平均值。

$$E_{ss} = \frac{1}{N} \sum_{k=N-m}^N |\psi_{ref}(k) - \psi(k)|$$

该指标反映了控制器抑制低频环境干扰的能力。

4. **控制输入平滑性**：这是衡量执行机构磨损程度的关键工程指标。定义为控制输入 δ 在整个仿真过程中的全变分之和。

$$TV = \sum_{k=1}^{N-1} |\delta(k+1) - \delta(k)|$$

TV 值越小，表明舵角的跳变和低频抖动越少，控制过程越平滑，舵机的能量消耗与机械磨损越低。在含有白噪声的系统中，该指标能有效区分不同控制策略对噪声的敏感程度。

3 控制器设计与实现

3.1 普通 PID 控制器设计

3.1.1 控制器结构与思路

控制器设计基于航向角误差 $e(t) = \psi_{ref}(t) - \psi(t)$ ，其连续时域形式描述如下：

$$u_{raw}(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt}$$

鉴于执行机构存在**物理幅度约束** $|\delta| \leq 0.5 \text{ rad}$ ，因此，本设计在标准 PID 基础上引入了**输出限幅**。

最终输出控制量 $\delta(t)$ 为：

$$\delta(t) = \text{sat}(u_{raw}(t), 0.5) = \begin{cases} 0.5, & u_{raw} > 0.5 \\ u_{raw}, & |u_{raw}| \leq 0.5 \\ -0.5, & u_{raw} < -0.5 \end{cases}$$

3.1.2 Simulink 实现

在本次实验中，我直接使用 Simulink 自带的 PID 模块，通过试凑法进行调参，最后得到参数如图2所示。

控制器参数	输出饱和
源: 内部	<input checked="" type="checkbox"/> 限制输出
比例 (P): 15	源: 内部
积分 (I): 0.001	上限: 0.5
微分 (D): 10	下限: -0.5
滤波器系数 (N): 100	<input type="checkbox"/> 线性化时忽略饱和和限值
	<input type="checkbox"/> 抗饱和

Figure 2: PID 参数设置

将 PID 控制器与系统进行连接，可以得到系统结构如图3所示。

其中 Monitor 模块负责将 ψ , ψ_{ref} 和 δ 的数据保存到工作区，以便后续对各项性能指标进行求取。USV 模块的结构如图4所示。

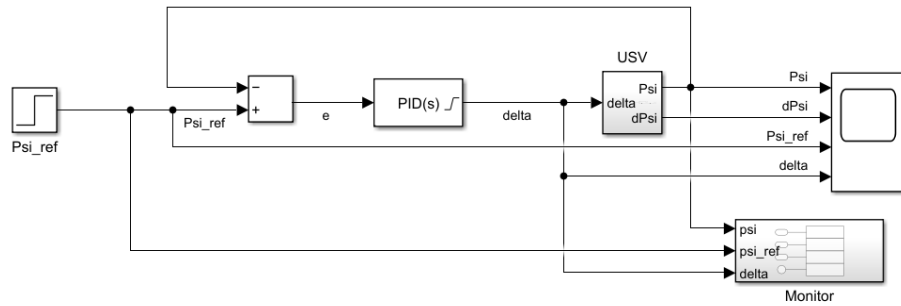


Figure 3: 普通 PID 系统

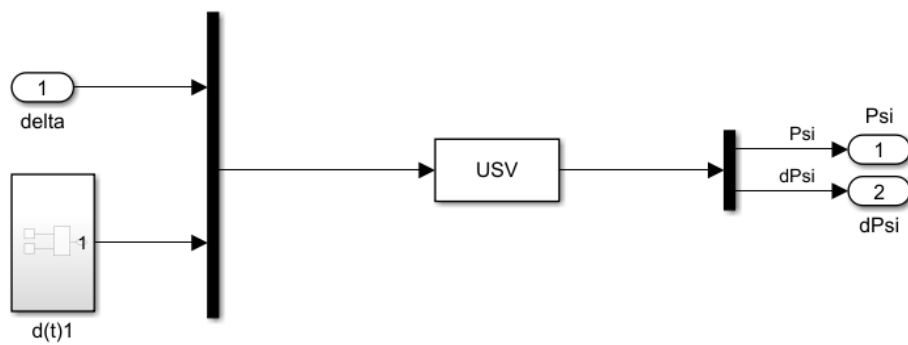


Figure 4: USV 模块

3.1.3 仿真结果

通过观察 scope 可以得到普通 PID 控制器的仿真结果如图5所示。

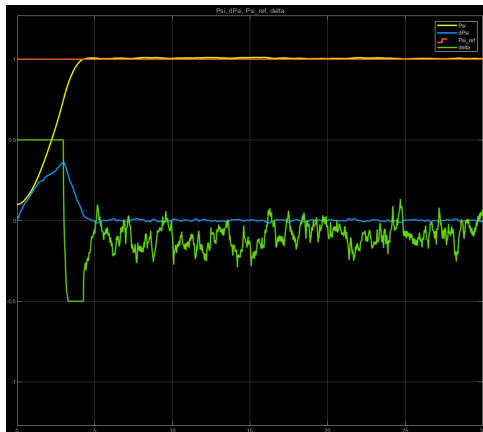


Figure 5: 普通 PID 仿真结果

其中黄色代表 ψ ，红色代表 ψ_{ref} ，蓝色代表 $\dot{\psi}$ ，绿色代表 δ 。

通过图像可以看出，在标准工况下，普通的 PID 控制器已经可以做到几乎无超调无稳态误差地快速跟踪 ψ_{ref} 。

3.2 直接专家控制器设计

为了克服传统控制系统“规则硬编码”导致的可维护性差、调试困难等问题，本设计采用了一种基于查表法的通用专家控制架构。该架构将专家系统的规则库和推理机进行完全分离设计。

3.2.1 规则库设计

通过将误差相平面 (e, ec) 划分为三个主要区域（饱和区、调节区、微调区），并结合误差变化率的方向，制定了以下五条核心控制规则。

1. 饱和控制规则：

- 当满足 $|e| > 0.7 \text{ rad}$ 时，系统采用饱和控制规则。
- 在该规则下，系统采用 **Bang-Bang 控制模式**，直接输出 δ 极限值 $\pm 0.5 \text{ rad}$ 。
- 当误差极大时，通过强制满舵，系统可以获得最大的回正力矩，从而减小调节时间。

2. 增强控制规则：

- 当满足 $0.2 < |e| \leq 0.7$ 且 $e \cdot ec > 0$ （误差与速度同向，即正在远离目标）时，系统采用增强控制规则。
- 在该规则下，系统输出**固定强控制量** $\pm 0.3 \text{ rad}$ 。
- 当船体处于中等偏差且有进一步恶化的趋势时，需要通过一个较强的输入量来遏制误差的发散，迫使 ec 变号。

3. 保守控制规则：

- 当满足 $0.2 < |e| \leq 0.7$ 且 $e \cdot ec \leq 0$ （误差正在减小，船正向目标靠拢）时，系统采用保守控制规则。
- 在该规则下，系统将**输出固定弱控制量** $\pm 0.1 \text{ rad}$ 。
- 此时系统误差已经在逐步缩小，因此需要通过一个较小的控制量，让系统依靠惯性逼近目标，从而显著减小超调量。

4. 精细控制规则：

- 当满足 $0.1 < |e| \leq 0.2$ 时，系统采用精细控制规则。
- 在该规则下，系统切换为 **PD 控制模式**，参数设定为参数设定为 $K_p = 25, K_d = 15$ 。
- 当误差较小，系统即将进入目标邻域时，采用 PD 控制可以确保系统平滑、无震荡地切入稳态。

5. 微调控制规则:

- 当满足 $|e| \leq 0.1$ 时, 系统采用微调控制规则。
- 在该规则下, 系统启用完整 PID 模式, 参数设定为 $K_p = 15, K_i = 0.25, K_d = 10$ 。
- 在稳态附近, USV 受到非线性水动力阻尼 $H(\dot{\psi})$ 和环境干扰 $d(t)$ 的持续作用, 纯 PD 控制无法消除静差。因此需要激活积分项, 从而消除最后的稳态误差, 实现高精度跟踪。

为了实现**规则与推理的彻底分离**, 所有的控制策略、阈值边界、增益参数均被封装在一个独立的 MATLAB 脚本文件 (Load_Expert_Rules.m) 中。其中, 知识被抽象为一个 $N \times 8$ 的二维矩阵 RuleMatrix。每一行代表一条完整的“IF-THEN”规则, 包含条件域 (误差 e 与变化率 ec 的上下限) 和动作域 (控制模式及 PID 参数), 具体结构如 Listing 2 所示。

该规则矩阵中, 每一列及其对应的信息有以下对应关系:

- **e 上下限**: 第一列为下限, 第二列为上限。
- **ec 上下限**: 第三列为下限, 第四列为上限。
- **输出模式**: 第五列代表输出模式, 1 为常量输出, 2 为 PID 输出。
- **输出量**: 在常量输出模式下, 第六列代表输出的值。
- **PID 参数**: 第六、七、八列分别代表 P、I、D 的系数。

Listing 2: 直接专家控制规则库

```

1 e1 = 0.7; % [大误差界限] 超过此值认为偏差极大, 需进入饱和控制区
2 e2 = 0.2; % [中误差界限] 区分强力调节区与精细调节区的边界
3 e3 = 0.1; % [小误差界限] 进入此范围认为已接近目标, 需启用微调
4
5 RuleMatrix = [
6     e1,  inf,  -inf,  inf,    1,    0.5,    0,    0;
7    -inf, -e1,  -inf,  inf,    1,   -0.5,    0,    0;
8     e2,   e1,    0,   inf,    1,    0.3,    0,    0;
9    -e1, -e2,  -inf,    0,    1,   -0.3,    0,    0;
10     e2,   e1,  -inf,    0,    1,    0.1,    0,    0;
11    -e1, -e2,    0,   inf,    1,   -0.1,    0,    0;
12     e3,   e2,  -inf,  inf,    2,   25.0,    0,   15;
13    -e2, -e3,  -inf,  inf,    2,   25.0,    0,   15;
14    -e3,   e3,  -inf,  inf,    2,   15.0,  0.25,   10;
15 ];

```

3.2.2 推理机设计

推理机是专家控制器的核心执行机构，其主要职责是将当前的系统状态映射为具体的控制动作。为了贯彻“知识与推理分离”的设计原则，本实验设计了一个**通用查表推理机**。该推理机不包含任何具体的控制参数（如控制输入幅值或 PID 增益），而是完全依赖外部输入的 RuleMatrix 进行决策，从而实现了算法逻辑的通用化。

推理机采用“**遍历匹配-优先级仲裁**”的机制运行，其内部逻辑流程如下：

1. **状态引入**：实时接收系统的误差 $e(t)$ 和误差变化率 $ec(t)$ 。
2. **规则遍历**：自上而下遍历规则矩阵 RuleMatrix。由于矩阵行序代表了规则的优先级（行号越小，优先级越高），一旦发现当前状态 (e, ec) 落在某行规则定义的条件区间 $[e_{min}, e_{max}] \cap [ec_{min}, ec_{max}]$ 内，立即判定匹配成功。
3. **模式解算**：根据匹配行第 5 列的 Mode 标志位，激活相应的控制算法：
 - **Mode 1 (固定输出/Bang-Bang)**：直接输出预设的控制量 U_{val} 。
 - **Mode 2 (线性调节/PID)**：读取 K_p, K_i, K_d 参数，结合状态记忆进行线性运算。
4. **安全输出**：如果遍历所有规则均未匹配（保底逻辑），输出 0；最后经过物理限幅环节（ $\pm 0.5 \text{ rad}$ ）输出最终舵角 δ 。

将直接专家控制器与系统进行连接，可以得到系统结构如图6所示。

其中 Cal_e 模块接收 ψ 和 ψ_{ref} 的输入，输出 e 和 ec 。

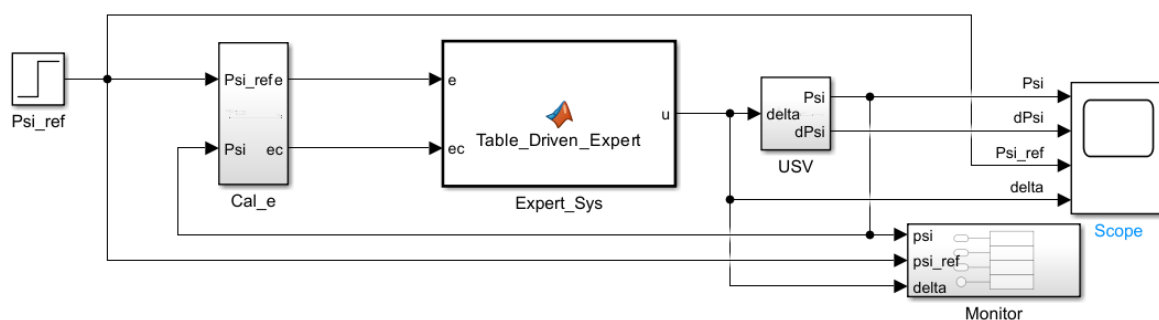


Figure 6: 直接专家控制系统

3.2.3 仿真结果

通过观察 scope 可以得到直接专家控制器的仿真结果如图7所示。

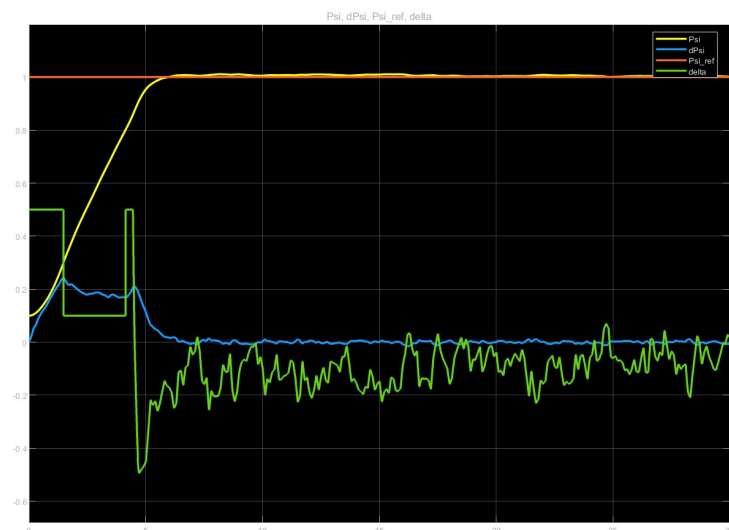


Figure 7: 直接专家控制仿真结果

通过图像可以看出，在标准工况下，直接专家控制系统也可以做到几乎无超调无稳态误差的快速跟踪。但与普通 PID 控制相比，调节时间较长。

3.3 专家 PID 控制器设计

与直接专家控制器的设计思路类似，专家 PID 控制依旧采用基于查表法的通用专家控制架构。

3.3.1 规则库设计

同样将误差相平面 (e, ec) 划分为三个主要区域，结合误差变化率的方向，制定了以下四条核心控制规则。

1. 抗饱和规则：

- 当满足 $|e| \geq 0.8$ 时，采用抗饱和控制规则。
- 在该规则下，系统采用 **Bang-Bang 控制模式**，直接输出 δ 极限值 ± 0.5 rad。
- 当误差极大时，直接采用 PID 会导致积分饱和，从而难以实现控制效果。

2. 加速响应规则：

- 当满足 $0.1 \leq |e| \leq 0.8$ 且 $e \cdot ec \geq 0$ 时，采用加速响应规则。
- 在该规则下，系统采用增强 P、D，削弱 I 的控制策略 ($K_p \times 7, K_i \times 0, K_d \times 5$)。
- 当误差增大时，通过增强 P、D 的作用来抑制偏差的增长，削弱 I 来防止积分饱和。

3. 抑制超调规则：

- 当满足 $0.1 \leq |e| \leq 0.8$ 且 $e \cdot ec \leq 0$ 时，采用抑制超调规则。
- 在该规则下，系统采用削弱 P，维持 D 的控制策略 ($K_p \times 0.7, K_i \times 0, K_d \times 1$)。
- 当误差减小时，通过削弱 P 来降低系统惯性冲量，保持 D 以维持阻尼作用，使得系统可以平滑过渡到目标值。

4. 消除静差规则：

- 当满足 $|e| \leq 0.1$ 时，采用消除静差规则。
- 在该规则下，系统采用削弱 D，增强 I 的控制策略 ($K_p \times 1, K_i \times 2.5, K_d \times 0.8$)。
- 当误差极小时，通过增强 I 来减小稳态误差，削弱 D 来防止白噪声引起的震荡。

与直接专家控制器的设计思路类似，所有的控制策略、阈值边界、增益参数均被封装在一个独立的 MATLAB 脚本文件 (Load_Expert_Rules.m) 中。其中，知识被抽象为一个 $N \times 9$ 的二维矩阵 ExpertPID_Matrix。具体结构如 Listing 3 所示。

该规则矩阵中，每一列及其对应的信息有以下对应关系：

- e 上下限：第一列为下限，第二列为上限。
- ec 上下限：第三列为下限，第四列为上限。
- 控制模式：第五列代表控制模式，1 为 Bang-Bang 控制，2 为 PID 控制。
- PID 参数：第六、七、八列分别代表 P、I、D 的系数。
- 输出量：在 Bang-Bang 控制下，第九列代表输出的值。

Listing 3: 专家 PID 控制规则库

```
1 Base_PID.Kp = 25; % [基准比例系数] 提供基础响应速度
2 Base_PID.Ki = 0.1; % [基准积分系数] 用于消除稳态误差
3 Base_PID.Kd = 15; % [基准微分系数] 提供基础阻尼
4
5 e1_PID = 0.8; % [抗饱和界限] 误差大于此值时，认为 PID 已无法有效
6 e2_PID = 0.1; % [精细调节界限] 误差小于此值时，进入稳态消除静差阶
7
8 ExpertPID_Matrix = [
9     e1_PID,    inf,  -inf,  inf,  1,   0,   0,   0,   0.5;
10    -inf, -e1_PID, -inf,  inf,  1,   0,   0,   0,  -0.5;
11     e2_PID,  e1_PID,   0,  inf,  2,  7.0, 0.0, 5.0,   0;
12    -e1_PID, -e2_PID, -inf,   0,  2,  7.0, 0.0, 5.0,   0;
13     e2_PID,  e1_PID, -inf,   0,  2,  0.7, 0.0, 1.0,   0;
14    -e1_PID, -e2_PID,   0,  inf,  2,  0.7, 0.0, 1.0,   0;
15    -e2_PID,  e2_PID, -inf,  inf,  2,   1, 2.5, 0.8,   0;
16 ];
```

3.3.2 推理机设计

专家 PID 控制器的推理机采用与直接专家控制器相同的架构,即通过“**遍历匹配-优先级仲裁**”的机制对规则进行匹配。匹配成功后,输出对应的 PID 参数或 Bang-Bang 控制量。

将专家 PID 控制器与系统进行连接,可以得到系统结构如图8所示。

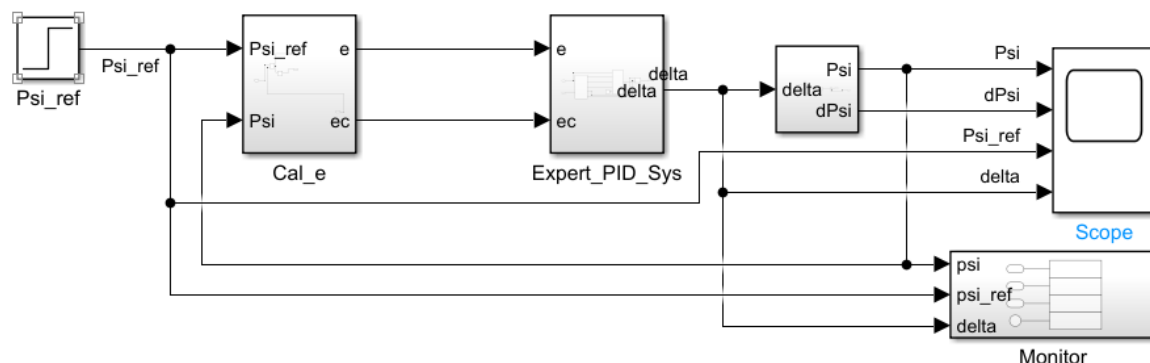


Figure 8: 专家 PID 控制系统

3.3.3 仿真结果

通过观察 scope 可以得到专家 PID 控制器的仿真结果如图9所示。

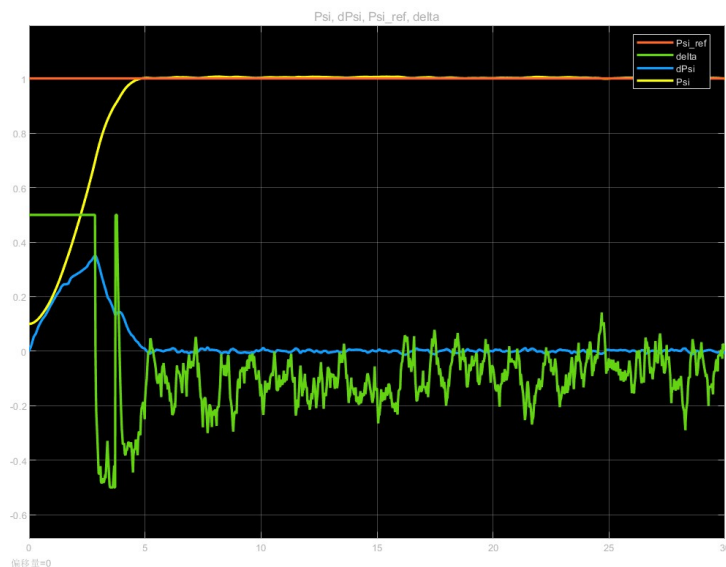


Figure 9: 专家 PID 控制仿真结果

通过图像可以看出,在标准工况下,专家 PID 控制系统也可以做到几乎无稳态误差无超调地快速跟踪目标值。通过性能指标的计算可以得出,专家 PID 控制系统具有最低的稳态误差和最低的超调量。

4 性能分析

为了全面评估三种控制器（普通 PID、直接专家控制、专家 PID）的性能，本章节基于 MATLAB/Simulink 仿真平台，在标称工况、参数摄动工况及强干扰工况下进行了系统的对比测试。

4.1 标准工况下的对比分析

在标准模型参数 ($T = 2.0, K = 0.8$) 及标准环境干扰下，对系统施加单位阶跃信号 ($\psi_{ref} = 1.0$)，三种控制器的性能指标对比如表1所示。

Table 1: 标准工况性能指标对比

控制器种类	超调量 (%)	调节时间 (s)	稳态误差	输入平滑性
普通 PID	1.42	3.73	0.0047	83.8655
直接专家控制器	1.44	4.97	0.0044	37.6679
专家 PID 控制器	1.00	4.00	0.0028	89.5189

仿真曲线对比如图10所示。其中绿色代表 ψ_{ref} ，红色代表普通 PID 控制器，黄色代表直接专家控制器，蓝色代表专家 PID 控制器。

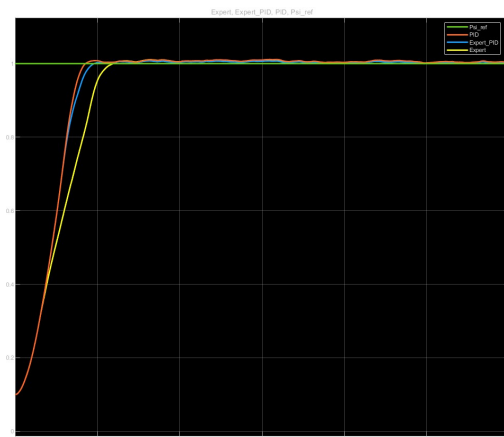


Figure 10: 标准工况仿真曲线图

结果分析：

1. **动态性能：**从调节时间看，普通 PID 控制器 (3.73s) 略优于专家 PID 控制器 (4s)，显著快于直接专家控制器 (4.97s)。

原因分析：普通 PID 控制器是线性连续控制器，在误差较大时能够持续输出以误差为比例的控制量。而直接专家控制器采用分段控制的策略，在“精细控制”和“保守控制”阶段采用了较小的增益或者固定的控制量，这种较为“保守”的策略虽然牺牲了部分响应速度，但是换取了更高的稳定性。

2. **稳态精度**：专家 PID 的稳态误差最小 (0.0028)，相比普通 PID 降低了约 40%。

原因分析：得益于专家 PID 的“消除静差规则”，在进入稳态区后，算法自动将积分增益 K_i 放大 (2.5 倍)，从而产生了比普通 PID 更强的静差消除能力。

3. **控制输入平滑性**：直接专家控制器具有远超其他两者的控制输入平滑性 ($TV = 37.6679$)。

原因分析：直接专家控制器在中间过渡过程采用了**固定输出控制**，避开了 PID 控制中微分项 D 对噪声的放大。

4.2 鲁棒性分析：参数摄动

为了验证控制器对模型不确定性的适应能力，分别模拟了“满载”（惯性增大）和“低速/舵效变差”（灵敏度降低）两种极端工况。

图11展示了极端工况下三种控制器的超调量对比；图12、13展示了两种工况下的仿真曲线。

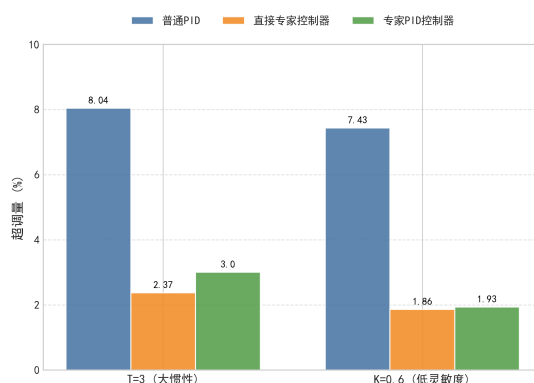


Figure 11: 超调量对比

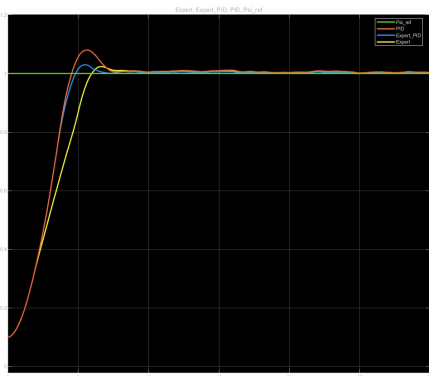


Figure 12: 大惯性 ($T = 3$)

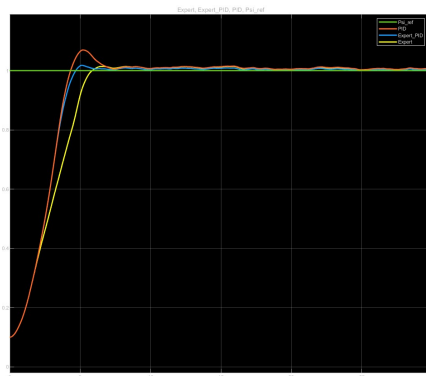


Figure 13: 低灵敏度 ($K = 0.6$)

结果分析：

1. **普通 PID**：改变 T 和 K 后，普通 PID 系统的超调量分别由标况下的 1.42% 激增到 8.04% 和 7.44%。

原因分析：固定参数的 PID 无法感知惯性的变化，原有的积分参数对于大惯性系统过强，从而导致大幅度超调。此外当灵敏度降低时，系统响应变慢，导致积分器在上升段积累了过多的误差，产生大幅度超调。

2. **专家系统**：从柱状图中可以看出，直接专家和专家 PID 控制器的超调量都能控制在 3% 以内，说明系统具有良好的稳定性。

原因分析：专家控制系统的控制逻辑是同时基于 e 和 ec 判断的。无论系统的参数如何变化，只要专家系统检测到误差在恶化就会改变控制器的参数，选择合适的控制方式。从参数摄动对比可以看出，**专家控制器拥有比普通 PID 控制器更好的鲁棒性**。

4.3 抗干扰性分析：环境噪声增强

为了比较不同控制器的抗干扰能力，将噪声的幅度提升到 0.3，模拟强干扰环境。

表2展示了强干扰工况下，三种控制器的各项性能指标；图14展示了强干扰工况下的仿真曲线。

Table 2: 强干扰工况性能指标对比

控制器	超调量 (%)	调节时间 (s)	稳态误差	输入平滑性
普通 PID	4.54	3.60	0.0141	83.7008
直接专家控制器	3.27	4.51	0.0120	37.4574
专家 PID 控制器	2.11	3.84	0.0078	90.7362

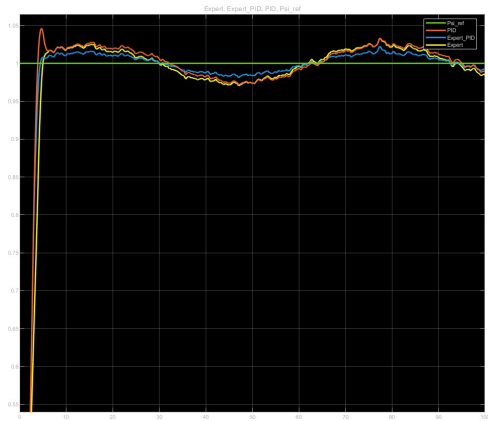


Figure 14: 强干扰工况仿真曲线

结果分析:

在仿真曲线中我对 y 轴进行放大, 可以看到专家 PID 控制器始终保持最贴近 ψ_{ref} , 拥有最低的稳态误差。

对比表中其他参数还可以发现, 在强干扰环境下, 两种专家控制器具有更低的超调量, 说明专家控制器具有比一般 PID 控制器更好的抗干扰能力。

4.4 跟踪性能分析: 正弦信号跟踪

设置参考航向为正弦信号 $\psi_{ref}(t) = \sin(0.2t)$, 测试系统的动态跟踪能力。

表3展示了正弦输入下的控制器性能指标; 图15展示了正弦输入下的仿真曲线。

Table 3: 不同控制器的跟踪误差与输入平滑性指标

控制器	最大跟踪误差 (rad)	平均跟踪误差 (rad)	输入平滑性
普通 PID	0.0306	0.0111	49.6038
直接专家控制器	0.0302	0.0113	38.2629
专家 PID 控制器	0.0203	0.0068	53.6593

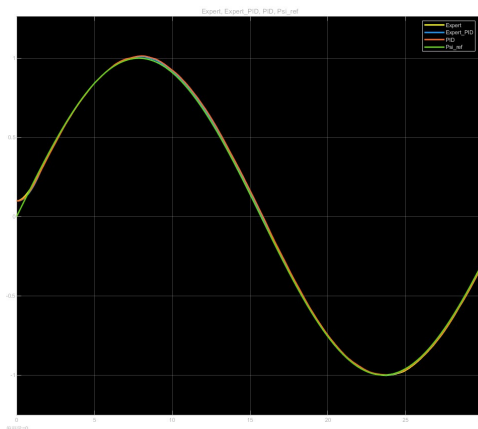


Figure 15: 正弦输入仿真曲线

结果分析:

观察仿真曲线可以看出, 四条曲线基本重合。因此对于正弦参考输入, 三种控制器均能达到较好的控制效果。

根据表中数据进行比较可知, **专家 PID 控制器具有最优跟踪能力**, **直接专家控制器具有最优输入平滑性**

5 结论

本次实验针对 USV 航向控制中存在的强非线性、大惯性等问题，设计并实现了普通 PID、直接专家控制以及专家 PID 控制系统，得出了以下结论：

1. **“规则分离”架构提升了系统的可扩展性：**实验中验证了独立规则库 + 查表法推理机的有效性。在实验调试过程中，这样的设计极大简化了参数整定流程。
2. **单一工况下普通 PID 具有较好效果：**根据实验仿真结果，在标准工况下，普通 PID 具有最短的调节时间，较低的超调量。因此，综合考虑成本和环境，**在工况变化不大的情况下**，可以直接选用普通 PID 进行控制。
3. **直接专家控制具有最优控制输入平滑性：**在本次实验的所有仿真结果中，直接专家控制器均具有最优控制输入平滑性，在其它性能指标方面也表现较为良好。因此若**任务对于控制输入平滑性，或控制量抖动带来的磨损有较高要求**，可以采用直接专家控制的方法。
4. **专家 PID 控制具有最优鲁棒性和控制精度：**在工况改变（如参数摄动、参考输入变化、噪声变化）时，专家 PID 控制均能取得最低的稳态误差和超调量，且调节时间短于直接专家控制。因此对于**工况变化较大，且对精度有较高要求**的场景，可以采用专家 PID 控制的方法。

6 心得体会

在本次实验中，我学习了使用 simulink 进行模型搭建，仿真调试。在搭建 USV 模型时，我额外学习了 S-Function 系统函数的方法，使用这种方法可以有效简化方块图的搭建。

在调试普通 PID 参数的时候，我尝试了试凑法，通过观察输出曲线形状、超调量、调节时间等特征，针对性地对系数进行调整。例如如果存在稳态误差可以增强积分，如果存在震荡可以增强微分，如果调节时间过长可以增强比例...

在设计两种专家控制器的时候，我首次尝试使用“规则推理分离”的程序设计方法。事实证明，这样的设计方法可以极大简化我后续对于参数的调试。

最后，在对三种不同控制器进行对比时，我也意识到每种控制器都有各自的优点和缺点，比如直接专家控制器调节时间较长，但是却具有最好的控制输入平滑性；专家 PID 控制器具有最好的鲁棒性和精度，但是频繁地切换 PID 控制模式会带来较大的抖动，控制输入平滑性最差。因此，要根据实际应用场景的特性选择合适的控制器。

A 附录 1：附件结构

```
src/
├── slprj/
├── Calc_Performance.m ..... 计算模型性能指标
├── compare.slx ..... 同时比较三个模型
├── compare.slxc
├── Load_Expert_PID_Rule.m ..... 专家 PID 控制器规则库
├── Load_Expert_Rule.m ..... 直接专家控制器规则库
├── Load_Noise.m ..... 噪声参数
├── params.mat ..... 工作台参数，如果无法正常运行模型加载这个
├── USV.m ..... USV 模型
├── USV_Expert.slx ..... 直接专家控制系统
├── USV_Expert.slxc
├── USV_Expert_PID.slx ..... 专家 PID 控制系统
├── USV_Expert_PID.slxc
├── USV_PID.slx ..... 普通 PID 控制系统
└── USV_PID.slxc
```

B 附录 2：推理机代码

Listing 4: 直接专家控制器推理机

```
1 function u = Table_Driven_Expert(e, ec, RuleMatrix)
2 % =====
3 % 函数名: Table_Driven_Expert (通用查表推理机)
4 % 功能: 根据输入的误差状态, 遍历规则矩阵, 执行对应的控制策略
5 % 输入:
6 %   e          - 当前航向误差 ( $\psi_{ref} - \psi$ )
7 %   ec         - 误差变化率 ( $d\psi/dt$ )
8 %   RuleMatrix - 从工作区传入的  $N \times 8$  规则矩阵
9 % 输出:
10 %   u          - 计算得到的控制量 (舵角  $\delta$ )
11 % =====
12
13 % --- 1. 输出初始化 ---
14 u = 0;
15
16 % 获取规则库的总规则数 (矩阵行数)
17 [NumRules, ~] = size(RuleMatrix);
18
19 % --- 2. 状态记忆变量定义 ---
```

```
20 % persistent 变量用于存储积分项，使其在函数调用间保持记忆
21 % 仅在规则 5（微调模式）中使用，用于消除稳态误差
22 persistent Integral_e;
23 if isempty(Integral_e)
24     Integral_e = 0;
25 end
26
27 % --- 3. 核心推理循环 (Inference Loop) ---
28 matched = false; % 初始化匹配标志位
29
30 for i = 1:NumRules
31     % 3.1 提取当前规则行的条件边界
32     e_min = RuleMatrix(i, 1);
33     e_max = RuleMatrix(i, 2);
34     ec_min = RuleMatrix(i, 3);
35     ec_max = RuleMatrix(i, 4);
36
37     % 3.2 逻辑判断：检查当前状态是否落在该规则的空间内
38     if (e >= e_min && e <= e_max) && (ec >= ec_min && ec <= ec_max)
39
40         % --- 3.3 匹配成功，提取控制参数 ---
41         Mode = RuleMatrix(i, 5); % 控制模式
42         Param1 = RuleMatrix(i, 6); % 参数1（固定值  $U$  或  $Kp$ ）
43         Param2 = RuleMatrix(i, 7); % 参数2 ( $Ki$ )
44         Param3 = RuleMatrix(i, 8); % 参数3 ( $Kd$ )
45
46         % --- 3.4 执行控制策略 ---
47         switch Mode
48             case 1 % === 模式 1: 固定输出控制 (Bang-Bang/Step) ===
49                 % 适用于饱和、增强、保守控制阶段
50                 % 直接输出预设的舵角值
51                 u = Param1;
52
53                 % 关键操作：退出线性区时清空积分器，防止积分饱和 (Anti-
54                     % windup)
55                 Integral_e = 0;
56
57             case 2 % === 模式 2: 线性 PID/PD 调节 ===
```

```
57         % 适用于精细控制和微调控制阶段
58
59         % 积分逻辑处理：仅当参数  $K_i$  ( $Param2$ ) 非零时累积积分
60         if Param2 ~= 0
61             Integral_e = Integral_e + e * 0.01; % 矩形积分法 ( $T_s = 0.01s$ )
62
63             % 积分限幅：防止稳态误差导致的积分过充
64             if abs(Integral_e) > 0.5
65                 Integral_e = sign(Integral_e) * 0.5;
66             end
67         else
68             % 若当前规则不需要积分 (如  $PD$  控制), 重置积分器
69             Integral_e = 0;
70         end
71
72         % 计算线性控制律:  $u = K_p * e + K_i * Integral + K_d * ec$ 
73         u = Param1 * e + Param2 * Integral_e + Param3 * ec;
74     end
75
76     matched = true;
77     break; % 优先级机制：找到第一条匹配规则后立即退出循环
78 end
79 end
80
81 % --- 4. 异常处理 (Safety Check) ---
82 % 如果遍历所有规则均未匹配 (理论上不应发生), 输出 0 舵角确保安全
83 if ~matched
84     u = 0;
85 end
86
87 % --- 5. 物理限幅 (Actuator Saturation) ---
88 % 模拟舵机的机械转角限制 ( $\pm 0.5 \text{ rad}$ )
89 if abs(u) > 0.5
90     u = sign(u) * 0.5;
91 end
92 end
```

Listing 5: 专家 PID 控制器推理机

```

1  function [Kp, Ki, Kd, Select_Flag, u_OL] = Expert_PID_Engine(e, ec,
    ExpertPID_Matrix, Base_PID)
2  % =====
3  % 函数名: Expert_PID_Engine (专家 PID 推理机)
4  % 功能: 基于当前状态查表, 输出 PID 参数乘数或开环控制指令
5  % 输入:
6  %   e           - 航向误差
7  %   ec          - 误差变化率
8  %   ExpertPID_Matrix - 专家规则矩阵 (来自工作区)
9  %   Base_PID      - 基准 PID 参数结构体
10 % 输出:
11 %   Kp, Ki, Kd     - 实时整定后的 PID 参数
12 %   Select_Flag   - 控制策略切换标志 (0=PID控制, 1=开环Bang-Bang)
13 %   u_OL          - 开环控制时的输出值
14 % =====
15
16 % --- 1. 初始化输出变量 ---
17 % 默认情况下, 使用基准 PID 参数, 且不启用开环控制
18 Kp = Base_PID.Kp;
19 Ki = Base_PID.Ki;
20 Kd = Base_PID.Kd;
21
22 Select_Flag = 0; % 默认使用 PID 回路
23 u_OL = 0;
24
25 % 获取规则库规模
26 [NumRules, ~] = size(ExpertPID_Matrix);
27 Matched = false; % 匹配标志位
28
29 % --- 2. 核心推理循环 (Rule Inference Loop) ---
30 for i = 1:NumRules
31     % 2.1 提取当前规则行的条件边界
32     e_min = ExpertPID_Matrix(i, 1);
33     e_max = ExpertPID_Matrix(i, 2);
34     ec_min = ExpertPID_Matrix(i, 3);
35     ec_max = ExpertPID_Matrix(i, 4);
36

```

```
37 % 2.2 逻辑判断：检查当前状态是否落在该规则定义区域内
38 if (e >= e_min && e <= e_max) && (ec >= ec_min && ec <= ec_max)
39
40 % --- 2.3 匹配成功，提取动作参数 ---
41 Mode = ExpertPID_Matrix(i, 5); % 控制模式
42 Mult_P = ExpertPID_Matrix(i, 6); % Kp 乘数
43 Mult_I = ExpertPID_Matrix(i, 7); % Ki 乘数
44 Mult_D = ExpertPID_Matrix(i, 8); % Kd 乘数
45 Val_OL = ExpertPID_Matrix(i, 9); % 开环输出值
46
47 % --- 2.4 执行控制动作 ---
48 if Mode == 1
49     % =====
50     % 模式 1: 开环 Bang-Bang 控制 (Open-Loop Control)
51     % 适用场景：规则 1 (抗饱和)
52     % 动作：强制切断 PID 回路，直接输出最大舵角
53     % =====
54     Select_Flag = 1; % 信号给 Switch 模块，切换到开环通道
55     u_OL = Val_OL; % 输出预设的满舵值 (+/- 0.5)
56
57     % 关键保护：将 PID 参数置零
58     % 原因：防止 PID 模块在后台继续计算误差积分，导致切回时出现
59     % 积分暴走 (Windup) 现象。
60     Kp = 0; Ki = 0; Kd = 0;
61
62 else
63     % =====
64     % 模式 2: 自适应 PID 参数整定 (Adaptive Tuning)
65     % 适用场景：规则 2, 3, 4 (加速、抑制、消差)
66     % 动作：基于基准参数乘以规则定义的因子，在线调整 PID
67     % =====
68     Select_Flag = 0; % 信号给 Switch 模块，使用 PID 通道
69     u_OL = 0; % 此值在 Select_Flag=0 时无效
70
71     % 参数计算公式：实时参数 = 基准值 * 乘数因子
72     Kp = Base_PID.Kp * Mult_P;
73     Ki = Base_PID.Ki * Mult_I;
74     Kd = Base_PID.Kd * Mult_D;
```

```
75         end
76
77         Matched = true;
78         break; % 优先级机制：命中高优先级规则后立即退出
79     end
80 end
81
82 % 如果未匹配任何规则(理论上不应发生)，保持默认基准参数输出
83 end
```