水下机器人推力分配矩阵计算原理

1. 基本概念

水下机器人推力分配矩阵（Thrust Allocation Matrix）$B \in \mathbb{R}^{6\times n}$ 建立了推进器推力向量 $\mathbf{f} \in \mathbb{R}^n$ 与机体坐标系下的广义力 $\boldsymbol{\tau} \in \mathbb{R}^6$ 之间的映射关系：

$$\boldsymbol{\tau} = B\mathbf{f} + \boldsymbol{\tau}_b(\boldsymbol{\eta})$$

其中：

- n 为推进器数量
- $\boldsymbol{\tau}_b$ 为浮力/重力产生的恢复力/力矩
- $\boldsymbol{\eta} = [x, y, z, \phi, \theta, \psi]^T$ 为位姿向量

2. 单推进器贡献模型

单个推进器的贡献由其在机体坐标系中的位置 $\mathbf{r}_i$ 和推力方向 $\mathbf{f}_i$ 决定：

$$\mathbf{b}_i = \begin{bmatrix} \mathbf{f}_i \\ \mathbf{r}_i \times \mathbf{f}_i \end{bmatrix} \in \mathbb{R}^6$$

其中：

- $\mathbf{f}_i = [f_{i_x}, f_{i_y}, f_{i_z}]^T$ 为单位推力方向向量
- $\mathbf{r}_i = [r_{i_x}, r_{i_y}, r_{i_z}]^T$ 为推进器位置

3. 推力分配矩阵构建

对于具有 n 个推进器的系统，B 矩阵为各推进器贡献向量的组合：

$$B = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_n \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1 & \mathbf{f}_2 & \cdots & \mathbf{f}_n \\ \mathbf{r}_1 \times \mathbf{f}_1 & \mathbf{r}_2 \times \mathbf{f}_2 & \cdots & \mathbf{r}_n \times \mathbf{f}_n \end{bmatrix}$$

4. 浮力/重力恢复力计算

浮力和重力产生的恢复力/力矩与姿态相关：

$$\boldsymbol{\tau}_b(\boldsymbol{\eta}) = \begin{bmatrix} \mathbf{f}_b \\ \mathbf{m}_b \end{bmatrix} = \begin{bmatrix} R_b^T(\mathbf{F}_g + \mathbf{F}_b) \\ R_b^T(\mathbf{r}_g \times \mathbf{F}_g + \mathbf{r}_b \times \mathbf{F}_b) \end{bmatrix}$$

其中：

- R_b 为从机体坐标系到世界坐标系的旋转矩阵
- $\mathbf{F}_g = [0, 0, -mg]^T$ 为重力
- $\mathbf{F}_b = [0, 0, \rho V g]^T$ 为浮力

- $\mathbf{r}_g, \mathbf{r}_b$ 分别为重心和浮心位置

5. 推力分配问题求解

5.1 伪逆法（Pseudo-inverse）

对于过驱动系统（$n > 6$），使用Moore-Penrose伪逆：

$$\mathbf{f} = B^{+}(\boldsymbol{\tau}_{des} - \boldsymbol{\tau}_b) = B^T(BB^T)^{-1}(\boldsymbol{\tau}_{des} - \boldsymbol{\tau}_b)$$

5.2 带约束的优化方法

考虑推进器饱和约束 $\mathbf{f}_{min} \leq \mathbf{f} \leq \mathbf{f}_{max}$，可构造二次规划问题：

$$\begin{aligned} \min_{\mathbf{f}} \quad & \frac{1}{2}\mathbf{f}^T W \mathbf{f} + \mathbf{c}^T \mathbf{f} \\ \text{s.t.} \quad & B\mathbf{f} = \boldsymbol{\tau}_{des} - \boldsymbol{\tau}_b \\ & \mathbf{f}_{min} \leq \mathbf{f} \leq \mathbf{f}_{max} \end{aligned}$$

其中 W 为权重矩阵，$\mathbf{c}$ 为成本向量。

6. 实现流程

```
def compute_thrust_allocation(eta, tau_des):
    # 1. 计算当前姿态下的B矩阵
    B = compute_thrust_matrix(eta)

    # 2. 计算浮力/重力补偿项
    tau_b = compute_buoyancy_compensation(eta)

    # 3. 求解推力分配问题
    if method == "pseudo_inverse":
        f = np.linalg.pinv(B) @ (tau_des - tau_b)
    elif method == "QP":
        f = solve_qp(B, tau_des - tau_b)

    # 4. 应用推力限幅
    f = np.clip(f, f_min, f_max)

    return f
```

7. 关键问题分析

7.1 奇异位形处理

当 $rank(B) < 6$ 时，系统处于奇异位形。解决方法包括：

- 奇异值分解（SVD）鲁棒伪逆
- 引入正则化项：$B^+ = B^T(BB^T + \lambda I)^{-1}$

7.2 能量最优分配

通过权重矩阵 W 实现能量最优分配：

$$W = diag\left(\frac{1}{T_1^{max}}, \cdots, \frac{1}{T_n^{max}}\right)$$

其中 T_i^{max} 为第 i 个推进器的最大推力。

8. 实例分析

对于URDF描述的6推进器系统，推力分配矩阵为：

$$B = \begin{bmatrix} 0 & 0 & -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ 0 & 0 & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -1 & -1 & 0 & 0 & 0 & 0 \\ -0.11 & 0.11 & 0.015\sqrt{2} & -0.015\sqrt{2} & -0.015\sqrt{2} & 0.015\sqrt{2} \\ 0.075 & 0.075 & 0.015\sqrt{2} & 0.015\sqrt{2} & -0.015\sqrt{2} & -0.015\sqrt{2} \\ 0 & 0 & -0.1746\frac{\sqrt{2}}{2} - 0.1093\frac{\sqrt{2}}{2} & \cdots & \cdots & \cdots \end{bmatrix}$$

9. 结论

水下机器人推力分配矩阵的计算需要综合考虑：

1. 推进器几何配置
2. 浮力/重力补偿
3. 系统约束条件
4. 能量优化目标

正确的推力分配算法能显著提高水下机器人的运动控制性能，特别是在存在强水流干扰的环境中。