5. SDK概述

AirbotPlay 机械臂控制 SDK 提供两套开发接口：

- **airbot_hardware**：底层硬件驱动库，提供机械臂电机的直控和反馈，用户可以使用此接口开发自己的运动控制算法、运动规划算法、数据采集和 VLA 算法推理等等
- **airbot_state_machine**：high level 上层应用开发接口，用户可以在机械臂预先安装的运动控制算法基础上，利用上层应用开发接口开发自己的软件业务功能，例　自主抓取等等

5.1 通讯架构图

下图呈现了开发者电脑与机械臂本体的系统组成及交互关系。开发者电脑部分涵盖运控算法节点和软件业务逻辑实现模块，通过上层应用开发接口airbot_state_machine和airbot_hardware的数据通讯控制机械臂本体的运动。机械臂本体由6个关节电机和夹爪组件构成，主控电脑负责软件运行。

```
graph LR
    subgraph "开发者电脑 (Developer PC)"
        ASM["airbot_state_machine<br/>运控算法节点<br/>+ 上层应用开发接口"]
        SDK["airbot_hardware SDK<br/>(C++11/Python)"]
    end

    subgraph "机械臂本体 (Robot Body)"
        MOTORS["关节电机 × 6<br/>Joint Motors"]
        GRIPPER["夹爪 × 1<br/>Gripper"]
        HAND["灵巧手 × 1<br/>Dexterous Hand"]
    end

    %% 连接关系
    ASM -.->|"控制命令"| SDK
    SDK -.->|"控制指令"| MOTORS
    SDK -.->|"控制指令"| GRIPPER
    SDK -.->|"控制指令"| HAND

    MOTORS -.->|"状态反馈"| SDK
    GRIPPER -.->|"状态反馈"| SDK
    HAND -.->|"状态反馈"| SDK
    SDK -.->|"状态信息"| ASM

    %% 样式定义
    classDef developer fill:#e1f5fe,stroke:#01579b,stroke-width:2px
    classDef hardware fill:#fff3e0,stroke:#e65100,stroke-width:2px

    class ASM,SDK developer
    class MOTORS,GRIPPER,HAND hardware
```

5.2 通讯流程说明

控制指令流程： 1. `airbot_state_machine` 生成控制指令

1. `airbot_hardware SDK` 转换为硬件控制命令
2. 控制关节电机、夹爪、灵巧手

状态反馈流程： 1. 硬件组件返回状态信息

1. `airbot_hardware SDK` 处理状态数据
2. `airbot_state_machine` 接收状态用于算法决策

5.3 支持的硬件配置

机械臂类型	电机配置	末端执行器	自由度
Play (6轴)	OD×3 + DM×3	无	6
Play (6轴)	OD×3 + ODM×3	无	6
PlayWithEEF (7轴)	OD×3 + DM×3	G2夹爪	7
PlayWithEEF (7轴)	OD×3 + DM×3	E2夹爪	7
Replay (7轴)	EC×6	E2夹爪	7

5.4 电机类型说明

电机类型	说明	适用关节
OD	求之自研一代电机	关节1-3
DM	达妙电机(新协议)	关节4-6
EC	编码器	所有关节

5.5 电机控制模式说明

控制模式	功能描述	适用场景
MIT	位置-速度-力矩复合控制，支持前馈力矩和PID参数调节	精确力控制、阻抗控制
PVT	位置-速度-时间控制，支持最大速度和力矩限制	轨迹跟踪、路径规划
CSV	纯速度控制	速度控制、连续运动
CSP	纯位置控制	位置控制、点到点运动

5.6 开发流程

1. **安装SDK包**（C++或Python）
2. **创建执行器实例**
3. **创建机械臂/电机实例**
4. **初始化硬件**
5. **启用硬件**
6. **发送控制命令**
7. **获取状态反馈**
8. **清理资源**