C++函数使用手册

Original Version 1.0.0

遨博（北京）智能科技有限公司

目录

1 简介

AUBO4.x API 是基于网络实现的，提供了大量用于操作机械臂的接口，包括登录与退出，;

本文件定义了使用机械臂的接口类. 是基于 C++开发的,其中类中的方法是操作机械臂的接口.

*Others: 接口中长度单位都为米(m)　角度单位（弧度）

*Function List:

1.1 回调函数的类型:

typedef void (*RealTimeJointStatusCallback)(aubo_robot_namespace::JointStatus *jointStatus, int size);　获取关节状态的回调函数类型

typedef void (*RealTimeRoadPointCallback) (const aubo_robot_namespace::wayPoint_S *wayPoint); 获取路点信息的回调函数类型

typedef void (*RealTimeEndSpeedCallback) (double speed); 获取末端速度的回调函数类型

typedef void (*RobotEventCallback) (const aubo_robot_namespace::RobotEventInfo *eventInfo); 获取机械臂事件信息的回调函数类型

1.2 构造函数

[redacted]ServiceInterface()

1.3 系统接口

[redacted]robotServiceLogin()

登录接口: 与机械臂服务器建立连接.　这个接口是使用其他所有接口的前提,只有在该接口正确返回的情况下,才能使 用其他接口.

robotServiceLogout()

退出登录,断开与机械臂服务器的连接

1.4 机械臂运动接口

robotServiceSetGlobalMoveMaxAcc() 设置全局的最大运动加速度

robotServiceSetGlobalMoveMaxVelc() 设置全局的最大运动速度

robotServiceJointMoveOnBaseCoord() 基于基座标系的关节运动 该函数进行了重载

robotServiceTeachJointMove() 示教关节运动

robotServiceTeachPosition() 示教位置运动

注：接口中关于长度的单位都为米 关于角度的单位都为弧度

2 数据类型

2.1 机械臂运动相关的类型

```
/** 机械臂状态枚举 **/
enum RobotState
{
    RobotStopped = 0,
    RobotRunning,
    RobotPaused,
    RobotResumed
};
/** 运动模式枚举 **/
enum move_mode
{
    NO_MOVEMODE = 0,
    MODEJ,
    MODEL,
    MODEP
};
/** 运动轨迹枚举 **/
enum move_track
{
    NO_TRACK = 0,
    //for moveJ and moveL
    TRACKING,
    //cartesian motion for movep
    ARC_CIR,
    CARTESIAN_MOVEP,
    CARTESIAN_CUBICSPLINE,
    CARTESIAN_UBSPLINEINTP,
    //joint motion    for movep
    JIONT_CUBICSPLINE,
    JOINT_UBSPLINEINTP,
};
/** 坐标系枚举 **/
enum coordinate_refer
{
    BaseCoordinate = 0,
    EndCoordinate,
    WorldCoordinate
};
/** 示教模式枚举 **/
```



```
enum teach_mode
{
    NO_TEACH = 0,
    JOINT1,
    JOINT2,
    JOINT3,
    JOINT4,
    JOINT5,
    JOINT6,
    MOV_X,
    MOV_Y,
    MOV_Z,
    ROT_X,
    ROT_Y,
    ROT_Z
};
/** 路点位置信息的表示方法 **/
struct Pos
{
    double x;
    double y;
    double z;
};
/** 路点位置信息的表示方法 **/
union cartesianPos_U
{
    Pos position;
    double positionVector[3];
};
/** 姿态的四元素表示方法 **/
struct Ori
{
    double w;
    double x;
    double y;
    double z;
};
/** 姿态的欧拉角表示方法 **/
struct Rpy
{
    double rx;
    double ry;
    double rz;
};
```

```
84.  /** 描述机械臂的路点信息 **/
85.  typedef struct
86.  {
87.      cartesianPos_U cartPos;       //机械臂的位置信息 X,Y,Z
88.      Ori orientation;              //机械臂姿态信息   通过四元素表示,可以通过工具函数实现与欧拉角互转
89.      double jointpos[ARM_DOF];     //机械臂关节角信息
90.  }wayPoint_S;
91.  /**
92.    * 描述关节的速度和加速度
93.    */
94.  typedef struct
95.  {
96.      double jointPara[ARM_DOF];
97.  }JointVelcAccParam;
98.  typedef struct
99.  {
100.     double jointPos[ARM_DOF];
101. }JointParam;
102. /**
103.   *  描述运动属性中的偏移属性
104.   */
105. typedef struct
106. {
107.     bool    ena;                  //是否使能偏移
108.     float relativePosition[3];    //偏移量 x,y,z
109. }MoveRelative;
```

2.2 机械臂关节状态

```
a.  /**
b.    * 该结构体描述机械臂的关节状态
c.    *
d.    */
e.  typedef struct PACKED
f.  {
g.      int      jointCurrentI;        /**< Current of driver     关节电流*/
h.      int      jointSpeedMoto;       /**< Speed of driver      关节速度*/
i.      float    jointPosJ;            /**< Current position in radian   关节角*/
j.      float    jointCurVol;          /**< Rated voltage of motor.Unit: mV   关节电压*/
k.      float    jointCurTemp;         /**< Current temprature of joint       当前温度*/
l.      int      jointTagCurrentI;     /**< Target current of motor           电机目标电流*/
m.      float    jointTagSpeedMoto;    /**< Target speed of motor             电机目标速度*/
n.      float    jointTagPosJ;         /**< Target position of joint in radian  目标关节角  */
o.      uint16 jointErrorNum;          /**< Joint error of joint num          关节错误码  */
```



```
}JointStatus;
```

2.3 事件类型

```
/**
 * event define    描述机械臂事件类型
 *
 * 机械臂的很多信息（故障，通知）是通过事件通知到客户的，所有在使用 SDK 时，
 * 务必注册接收事件的回调函数。
 */
typedef enum{
    RobotEvent_armCanbusError,              //机械臂 CAN 总线错误
    RobotEvent_remoteHalt,                  //远程关机      TODO
    RobotEvent_remoteEmergencyStop,         //机械臂远程急停
    RobotEvent_jointError,                  //关节错误
    RobotEvent_forceControl,                //力控制
    RobotEvent_exitForceControl,            //退出力控制
    RobotEvent_softEmergency,               //软急停
    RobotEvent_exitSoftEmergency,           //退出软急停
    RobotEvent_collision,                   //碰撞
    RobotEvent_collisionStatusChanged,      //碰撞状态改变
    RobotEvent_tcpParametersSucc,           //工具动力学参数设置成功
    RobotEvent_powerChanged,                //机械臂电源开关状态改变
    RobotEvent_ArmPowerOff,                 //机械臂电源关闭
    RobotEvent_mountingPoseChanged,         //安装位置发生改变
    RobotEvent_encoderError,                //编码器错误
    RobotEvent_encoderLinesError,           //编码器线数不一致
    RobotEvent_singularityOverspeed,        //奇异点超速
    RobotEvent_currentAlarm,                //机械臂电流异常
    RobotEvent_toolioError,                 //机械臂工具端错误
    RobotEvent_robotStartupPhase,           //机械臂启动阶段
    RobotEvent_robotStartupDoneResult,      //机械臂启动完成结果
    RobotEvent_robotShutdownDone,           //机械臂关机结果
    RobotEvent_atTrackTargetPos,            //机械臂轨迹运动到位信号通知
    RobotSetPowerOnDone,                    //设置电源状态完成
    RobotReleaseBrakeDone,                  //机械臂刹车释放完成
    RobotEvent_robotControllerStateChaned,  //机械臂控制状态改变
    RobotEvent_robotControllerError,        //机械臂控制错误----一般是算法规划出现问题时返回
    RobotEvent_socketDisconnected,          //socket 断开连接
    RobotEvent_robotControlException,
    RobotEvent_exceptEvent = 100,
    //unknown event
    robot_event_unknown,
```

```
    //user event
    RobotEvent_User = 1000,                    // first user event id
    RobotEvent_MaxUser = 65535                 // last user event id
}RobotEventType;
```

```
/** 事件类型 **/
typedef struct{
    RobotEventType  eventType;      //事件类型号
    int             eventCode;      //
    std::string     eventContent;   //事件内容
}RobotEventInfo;
```

2.4 诊断信息

```
/****机械臂诊断信息****/
typedef struct PACKED
{
    //CAN 通信状态:0x01~0x80：关节 CAN 通信错误（每个关节占用 1bit）
    //0x00：无错误 0xff：CAN 总线存在错误
    uint8 armCanbusStatus;
    //机械臂 48V 电源当前电流
    float armPowerCurrent;
    //机械臂 48V 电源当前电压
    float armPowerVoltage;
    //机械臂 48V 电源状态（开、关）
    bool   armPowerStatus;
    //控制箱温度
    char   contorllerTemp;
    //控制箱湿度
    uint8 contorllerHumidity;
    //远程关机信号
    bool   remoteHalt;
    //机械臂软急停
    bool   softEmergency;
    //远程急停信号
    bool   remoteEmergency;
    //碰撞检测位
```



```
x.          bool    robotCollision;
y.          //机械臂进入力控模式标志位
z.          bool    forceControlMode;
aa.         //刹车状态
ab.         bool brakeStuats;
ac.         //末端速度
ad.         float robotEndSpeed;
ae.         //最大加速度
af.         int robotMaxAcc;
ag.         //上位机软件状态位
ah.         bool orpeStatus;
ai.         //位姿读取使能位
aj.         bool enableReadPose;
ak.         //安装位置状态
al.         bool robotMountingPoseChanged;
am.         //磁编码器错误状态
an.         bool encoderErrorStatus;
ao.         //静止碰撞检测开关
ap.         bool staticCollisionDetect;
aq.         //关节碰撞检测  每个关节占用 1bit 0-无碰撞  1-存在碰撞
ar.         uint8 jointCollisionDetect;
as.         //光电编码器不一致错误  0-无错误  1-有错误
at.         bool encoderLinesError;
au.         //joint error status
av.         bool jointErrorStatus;
aw.         //机械臂奇异点过速警告
ax.         bool singularityOverSpeedAlarm;
ay.         //机械臂电流错误警告
az.         bool robotCurrentAlarm;
ba.         //tool error
bb.         uint8 toolIoError;
bc.         //机械臂安装位置错位（只在力控模式下起作用）
bd.         bool robotMountingPoseWarning;
be.         //mac 缓冲器长度
bf.         uint16 macTargetPosBufferSize;
bg.         //mac 缓冲器有效数据长度
bh.         uint16 macTargetPosDataSize;
bi.         //mac 数据中断
bj.         uint8    macDataInterruptWarning;
bk.     }RobotDiagnosis;
```

2.5 事件相关的回调函数类型

```
/**
 * @brief 获取实时关节状态的回调函数类型
 * @param jointStatus 当前的关节状态;
 * @param size 上一个参数（jointStatus）的长度;
 * @param arg 这个参数是使用者在注册回调函数中传递的第二个参数;
 */
typedef void (*RealTimeJointStatusCallback)(const aubo_robot_namespace::JointStatus *jointStatus, int size, void *arg);
/**
 * @brief 获取实时路点信息的回调函数类型
 * @param wayPoint 当前的路点信息;
 * @param arg 这个参数是使用者在注册回调函数中传递的第二个参数;
 */
typedef void (*RealTimeRoadPointCallback) (const aubo_robot_namespace::wayPoint_S *wayPoint, void *arg);
/**
 *@brief 获取实时末端速度的回调函数类型
 *@param speed 当前的末端速度;
 *@param arg 这个参数是使用者在注册回调函数中传递的第二个参数;
 */
typedef void (*RealTimeEndSpeedCallback) (double speed, void *arg);
/**
 * @brief 获取机械臂事件信息的回调函数类型
 * @param arg 这个参数是使用者在注册回调函数中传递的第二个参数;
 */
typedef void (*RobotEventCallback) (const aubo_robot_namespace::RobotEventInfo *eventInfo, void *arg);
```

2.6 工具参数类型

[redacted]工具运动学参数

```
/** 该结构体描述工具的参数
  * 工具标定后既有相对末端坐标系的位置也有姿态。
  */
typedef struct
{
    Pos toolInEndPosition;    //工具相对末端坐标系的位置
    Ori toolInEndOrientation; //工具相对末端坐标系的姿态
}ToolInEndDesc;

typedef ToolInEndDesc ToolKinematicsParam;  //运动学参数
```


该参数主要描述工具的相对于机械臂末端的位置和姿态。ToolInEndDesc 类型用于描述工具标定的结果；ToolKinematicsParam 常用于带工具运动设置运动学参数。

工具动力学参数

```
1.  /** 该结构体描述工具惯量 **/
2.  typedef struct
3.  {
4.      double xx;
5.      double xy;
6.      double xz;
7.      double yy;
8.      double yz;
9.      double zz;
10. }ToolInertia;

11. /**
12.  * 该结构体描述工具的 动力学参数
13.  *
14.  * 注意：在跟换机械臂的工具时，工具的动力学参数和运动学参数是需要一起设置的。
15.  **/
16. typedef struct
17. {
18.     double positionX;     //工具重心的 X 坐标
19.     double positionY;     //工具重心的 Y 坐标
20.     double positionZ;     //工具重心的 Z 坐标
21.     double payload;        //工具重量
22.     ToolInertia toolInertia;  //工具惯量    待开发
23. }ToolDynamicsParam;
```

该类型 ToolDynamicsParam 用于描述工具的动力学参数。

2.7 坐标系标定

```
a.  /** 坐标系标定方法枚举 **/
b.  enum CoordCalibrateMathod
c.  {
d.      Origin_AnyPointOnPositiveXAxis_AnyPointOnPositiveYAxis,                // 原点、x 轴正半轴、y 轴正半轴
e.      Origin_AnyPointOnPositiveYAxis_AnyPointOnPositiveZAxis,                // 原点、y 轴正半轴、z 轴正半轴
f.      Origin_AnyPointOnPositiveZAxis_AnyPointOnPositiveXAxis,                // 原点、z 轴正半轴、x 轴正半轴
g.      Origin_AnyPointOnPositiveXAxis_AnyPointOnFirstQuadrantOfXOYPlane,  // 原点、x 轴正半轴、x、y 轴平面的第一象限上任意一点
h.      Origin_AnyPointOnPositiveXAxis_AnyPointOnFirstQuadrantOfXOZPlane,  // 原点、x 轴正半轴、x、z 轴平面的第一象限上任意一点
```

```
        Origin_AnyPointOnPositiveYAxis_AnyPointOnFirstQuadrantOfYOZPlane,   // 原点、y 轴正半轴、y、z 轴平面的第一象限上任意一点
        Origin_AnyPointOnPositiveYAxis_AnyPointOnFirstQuadrantOfYOXPlane,   // 原点、y 轴正半轴、y、x 轴平面的第一象限上任意一点
        Origin_AnyPointOnPositiveZAxis_AnyPointOnFirstQuadrantOfZOXPlane,   // 原点、z 轴正半轴、z、x 轴平面的第一象限上任意一点
        Origin_AnyPointOnPositiveZAxis_AnyPointOnFirstQuadrantOfZOYPlane,   // 原点、z 轴正半轴、z、y 轴平面的第一象限上任意一点
        CoordTypeCount
    };
    /**
     * 该结构体描述一个坐标系。系统根据该结构体能够唯一个确定一个坐标系。
     *
     * 坐标系分 3 种类型：基座标系(BaseCoordinate)，末端坐标系(EndCoordinate)，用户坐标系(WorldCoordinate);
     *
     * 基座坐标系是 根据机械臂底座建立的坐标系
     * 末端坐标系是 根据法兰盘中心建立的坐标系
     * 用户坐标系是 用户根据自己的需求建立的实际需要用到的坐标系,系统根据用户提供的 3 个点和标定方法确定用户坐标系的 X 轴,Y 轴,Z 轴。
     *          在实际应用中标定坐标系时会用到工具，系统为了准确的得到坐标系标定的 3 个点，所以用户需要提供工具信息，
     *          如果没有使用工具可以将工具描述(toolDesc)设置为 0.
     *
     * 使用说明：
     *    1:当 coordType==BaseCoordinate 或者 coordType==EndCoordinate 时，下面 3 个参数(methods,wayPointArray,toolDesc)系统不做处理，
     *     因为系统根据这个参数 coordType 已经可以确定该坐标系了。
     *
     *    2:如果坐标系标定的时候没有使用工具，工具描述中的位置和姿态信息应设置为 0。
     */
    typedef struct
    {
        coordinate_refer      coordType;          //坐标系类型：当 coordType==BaseCoordinate 或者 coordType==EndCoordinate 是，下面 3 个参数不做处理
        CoordCalibrateMathod methods;             // 坐标系标定方法
        JointParam           wayPointArray[3];    //用于标定坐标系的 3 个点（关节角），对应于机械臂法兰盘中心点基于基座标系
        ToolInEndDesc        toolDesc;            //标定的时候使用的工具描述
    }CoordCalibrateByJointAngleAndTool;
```

2.8 IO 相关数据类型

```
    /**
```



```
 * 描述 IO 的类型
 **/
typedef enum
{
    RobotBoardControllerDI,     //接口板控制器 DI(数字量输入)    只读(一般系统内部使用)
    RobotBoardControllerDO,     //接口板控制器 DO(数字量输出)    只读(一般系统内部使用)
    RobotBoardControllerAI,     //接口板控制器 AI(模拟量输入)    只读(一般系统内部使用)
    RobotBoardControllerAO,     //接口板控制器 AO(模拟量输出)    只读(一般系统内部使用)
    RobotBoardUserDI,           //接口板用户 DI(数字量输入)      可读可写
    RobotBoardUserDO,           //接口板用户 DO(数字量输出)      可读可写
    RobotBoardUserAI,           //接口板用户 AI(模拟量输入)      可读可写
    RobotBoardUserAO,           //接口板用户 AO(模拟量输出)      可读可写
    RobotToolDI,                //工具端 DI
    RobotToolDO,                //工具端 DO
    RobotToolAI,                //工具端 AI
    RobotToolAO,                //工具端 AO
}RobotIoType;
```

```
/**
 * 综合描述一个 IO
 **/
typedef struct PACKED
{
    char            ioId[32];       //IO-ID 目前未使用
    RobotIoType ioType;             //IO 类型
    char            ioName[32];     //IO 名称
    int             ioAddr;         //IO 地址
    double          ioValue;        //IO 状态
}RobotIoDesc;
/**
 * 工具的电源类型
 **/
typedef enum
{
    OUT_0V  = 0,
    OUT_12V = 1,
    OUT_24V = 2
}ToolPowerType;
typedef  enum                   //IO 状态
{
    IO_STATUS_INVALID = 0, //有效
    IO_STATUS_VALID         //无效
}IO_STATUS;
```

3 API

3.1 API 之连接与断开模块

概述：

本部分接口包括与机械臂服务器建立网络连接、断开连接、查看当前的连接状态；与服务器成功建立网络连接是使用其他接口的前提。接口实现使用的是 TCP/IP 协议，因此成功建立连接是使用其他接口的前提。

[redacted]登录接口 (与机械臂服务器建立网络连接)

```
a.  int  robotServiceLogin(const char* host, int port, const char *userName, const char* possword);
```

函数功能：该函数用于与机械臂服务器建立网络连接，该函数调用成功，是使用其他接口的前提。

参数描述：1、host 机械臂服务器 IP 地址，即控制器的 IP;

2、port 默认为 8899

3、userName 用户名默认为：AUBO

4、possword 密码默认为:123456

返回值：调用成功返回 ErrnoSucc;错误返回错误号

[redacted]退出登录接口

```
a.  int  robotServiceLogout();
```

函数功能：该函数用于与机械臂服务器断开连接

返回值：调用成功返回 ErrnoSucc;错误返回错误号

[redacted]当前连接状态接口

```
a.  void robotServiceGetConnectStatus(bool &connectStatus);
```

函数功能：该函数用于查看与机械臂服务器的连接状态

参数描述：connectStatus 是一个传出参数，网络处于连通状态返回 true;否则返回 false.

3.2 API 之机械臂初始化模块

概述：

机械臂的初始化与关闭；初始化会完成对机械臂的上电，松刹车等 chazuo

[redacted]机械臂启动

```
a.  int rootServiceRobotStartup( const aubo_robot_namespace::ToolDynamicsParam &toolDynamicsParam,
b.                               uint8 collisionClass,
c.                               bool readPose,
d.                               bool staticCollisionDetect,
e.                               int boardBaxAcc,
f.                               aubo_robot_namespace::ROBOT_SERVICE_STATE &result,
g.                               bool IsBolck = true);
```

函数功能：该函数用于启动机械臂，包括上电，松刹车，设置碰撞等级，设置动力学参数等，该函数完成需要的时间比较长，因此可以通过设置是否阻塞来调整函数返回的时间，当设置为非阻塞是，函数调用后立即返回，调用结果通过事件来通知。当设置为阻塞时，参数返回值代表接口是否被调用成功，result 表示机械臂启动结果。

参数描述：1、toolDynamicsParam 为工具的动力学参数，如果末端夹持工具，此参数应该根据具体的来设定；如果末端未夹持工具，将此参数的各项设置为 0；

2、collisionClass 机械臂的碰撞等级

3、readPose 是否允许接口读取位姿 默认设置为 true

4、staticCollisionDetect 默认设置为 true

5、result 机械臂启动结果只有 result==ROBOT_SERVICE_WORKING 表示机械臂启动成功，否则表示启动失败。

6、IsBolck 调用接口时是否阻塞

返回值：调用成功返回 ErrnoSucc;错误返回错误号

[redacted]机械臂关机

```
a.  int    rootServiceRobotShutdown(bool IsBolck = true);
```

函数功能：机械臂关机

参数描述：IsBolck 调用接口时是否阻塞：设置阻塞时，直到机械臂关机后函数返回；设置非阻塞时，立即返回，结果通过事件推送来返回。

返回值：调用成功返回 ErrnoSucc;错误返回错误号

3.3 API 之信息推送

概述：

本部分接口主要是关于机械臂信息推送的。接口中机械臂的信息推送是通过回调函数来实现的，因此要实现信息推送，开发者必须自己定义回调函数，并使用下面接口将自己定义的回调函数注册到系统中，这样方可实现信息推送。本部分接口中包含关于实时关节信息的推送，实时路点信息的推送，实时末端速度的推送，机械臂事件的推送；下面接口用于将用户定义的回调函数注册到我们系统中。从而实现信息推送。

实时关节信息的推送

```
1.  int  robotServiceRegisterRealTimeJointStatusCallback(RealTimeJointStatusCallback ptr, void  *arg);
```

函数功能：该函数用于注册获取“实时关节信息的回调函数”到系统，成功注册后,服务器会通过回调函数实时推送当前的关节状态信息.

参数描述：1、ptr 为获取实时关节状态信息的回调函数指针，当 ptr==NULL 时，相当于取消回调函数的注册;

2、这个参数系统不做任何处理，只是进行缓存，当系统调用已注册回调函数时该参数会通过回调函数的参数传回。

返回值：调用成功返回 ErrnoSucc;错误返回错误号

实时路点信息的推送

```
1.  int  robotServiceRegisterRealTimeRoadPointCallback(const RealTimeRoadPointCallback ptr, void  *arg);
```

函数功能：该函数用于注册获取“实时路点信息的回调函数”到系统，成功注册后,服务器会通过回调函数实时推送机械臂当前路点信息.

参数描述：1、ptr 为获取实时路点信息的回调函数指针，当 ptr==NULL 时，相当于取消回调函数的注册;

2、这个参数系统不做任何处理，只是进行缓存，当系统调用已注册回调函数时该参数会通过回调函数的参数传回。

返回值：调用成功返回 ErrnoSucc;错误返回错误号

实时末端速度

```
1.  int  robotServiceRegisterRealTimeRoadPointCallback(const RealTimeRoadPointCallback ptr, void  *arg);
```


函数功能：该函数用于注册获取“实时末端速度的回调函数”到系统，成功注册后,服务器会通过回调函数实时推送末端速度信息.

参数描述：1、ptr 为获取末端速度信息的回调函数指针，当 ptr==NULL 时，相当于取消回调函数的注册;

2、这个参数系统不做任何处理，只是进行缓存，当系统调用已注册回调函数时该参数会通过回调函数的参数传回。

返回值：调用成功返回 ErrnoSucc;错误返回错误号

机械臂事件信息

```
1.  int  robotServiceRegisterRobotEventInfoCallback(RobotEventCallback ptr, void  *arg);
```

函数功能：该函数用于注册获取“的回调函数”到系统，成功注册后,服务器会通过回调函数实时推送事件信息.

参数描述：1、ptr 为获取事件信息的回调函数指针，当 ptr==NULL 时，相当于取消回调函数的注册;

2、这个参数系统不做任何处理，只是进行缓存，当系统调用已注册回调函数时该参数会通过回调函数的参数传回。

返回值：调用成功返回 ErrnoSucc;错误返回错误号

备注：机械臂的大部分通知类事件和故障信息都是通过事件推送的。因此该函数非常重要。

关于信息推送的示例代码，详见附录.

3.4 API 之运动模块

概述：

本部分接口是关于运动相关的接口，包括运动属性的设置和多种形式的运动。在运动之前应该设置好对用的运动属性。

运动属性包括:

1.关节型运动的最大速度和最大加速度，默认每个关节的最大加速度为 25 度每秒方，默认每个关节的最大速度为 25 度每秒，该属性在关节型运动中生效；

2.末端型运动的最大线速度和最大线加速度，默认最大线加速度为 0.03 米每秒方，默认最大线速度为 0.03 米每秒，该属性在末端型运动中生效；

3.末端型运动的最大角速度和最大角加速度，默认最大角加速度为 100 度每秒方，默认最大角速度为 100 度每秒，该属性在末端型运动中生效；

4.轨迹运动的路点容器，默认容器为空，该属性在非示教运动中的圆轨迹运动中生效；

5.轨迹运动的交融半径，默认交融半径为 0，当轨迹运动类型==CARTESIAN_MOVEP(MOVEP 轨迹)时，该参数有效；

6.轨迹运动中圆轨迹的圈数，默认为 0，当轨迹运动类型等于 ARC_CIR 时该属性生效；且圆轨迹的圈数等于零时，ARC_CIR 代表圆弧，圆轨迹的圈数大于零时，ARC_CIR 代表圆；

7.运动属性至偏移量属性，默认为无偏移，该属性在非示教运动中生效；

8.设置示教运动的坐标系，默认为基座标系，该属性只在示教运动中生效；

运动接口：

关节运动（robotServiceJointMove）:通过关节运动的形式运动至目标位置。

直线运动（robotServiceLineMove）:以直线运动的形式运动至目标位置。

旋转运动（robotServiceRotateMove）:保持位置不变，做姿态的变换

轨迹运动（robotServiceTrackMove）:根据轨迹的类型进行运动

直线运动至目标位置(robotMoveLineToTargetPosition):保持当前姿态不变以直线运动的形式运动至目标位置。

关节运动至目标位置(robotMoveJointToTargetPosition):保持当前姿态不变以直线运动的形式运动至目标位置。

初始化运动属性

```
a.  int  robotServiceInitGlobalMoveProfile();
```

函数功能：对运动属性进行初始化，将个属性设置为初始值。

运动属性默认值描述:

1.关节型运动的最大速度和最大加速度，默认每个关节的最大加速度为 25 度每秒方，默认每个关节的最大速度为 25 度每秒；

2.末端型运动的最大线速度和最大线加速度，默认最大线加速度为 0.03 米每秒方，默认最大线速度为 0.03 米每秒；

3.末端型运动的最大角速度和最大角加速度，默认最大角加速度为 100 度每秒方，默认最大角速度为 100 度每秒；

4.轨迹运动的路点容器，默认容器为空；

5.轨迹运动的交融半径，默认交融半径为 0；

6.轨迹运动中圆轨迹的圈数，默认为 0；

7.运动属性至偏移量属性，默认为无偏移；

8.设置示教运动的坐标系，默认为基座标系；

返回值：调用成功返回 ErrnoSucc;错误返回错误号

设置和获取机械臂运动属性中关节型运动的最大速度和最大加速度

```
a.  int   robotServiceSetGlobalMoveJointMaxAcc (const aubo_robot_namespace::JointVelcAccParam   &moveMaxAcc);
b.
c.  int   robotServiceSetGlobalMoveJointMaxVelc(const aubo_robot_namespace::JointVelcAccParam   &moveMaxVelc);
d.
e.  void  robotServiceGetGlobalMoveJointMaxAcc (aubo_robot_namespace::JointVelcAccParam         &moveMaxAcc);
f.
g.  void  robotServiceGetGlobalMoveJointMaxVelc(aubo_robot_namespace::JointVelcAccParam         &moveMaxVelc);
```

机械臂的运动分为关节型运动和末端型运动，上面四个接口主要用于设置关节型运动的最大速度和最大加速度。关节型运动需要设置每个关节的大速度和最大加速度。最大速度不超过 180 度每秒,最大加速度不超过 180 度每秒方。

注意：接口中涉及的的角度单位是弧度，长度单位是米。

设置和获取机械臂运动属性中末端型运动的最大速度和最大加速度

```
a.  int   robotServiceSetGlobalMoveEndMaxLineAcc (double   moveMaxAcc);
b.
c.  int   robotServiceSetGlobalMoveEndMaxLineVelc(double   moveMaxVelc);
d.
e.  void  robotServiceGetGlobalMoveEndMaxLineAcc (double   &moveMaxAcc);
f.
g.  void  robotServiceGetGlobalMoveEndMaxLineVelc(double   &moveMaxVelc);
h.
i.  int   robotServiceSetGlobalMoveEndMaxAngleAcc (double   moveMaxAcc);
j.
k.  int   robotServiceSetGlobalMoveEndMaxAngleVelc(double   moveMaxVelc);
l.
m.  void  robotServiceGetGlobalMoveEndMaxAngleAcc (double   &moveMaxAcc);
```

机械臂的运动分为关节型运动和末端型运动，本部分接口主要用于设置末端型运动的最大线速度、最大线加速度、最大角速度、最大角加速度。最大线速度不超过 3 米每秒，最大线加速度不超过 3 米每秒方，最大加速度不超过 3 米每秒方；最大角速度不超过 180 度每秒,最大角加速度不超过 180 度每秒方。

注意：接口中涉及的的角度单位是弧度，长度单位是米。

对轨迹运动中路点集合的添加，删除，获取

```
a.  void  robotServiceClearGlobalWayPointVector();
b.
c.  int   robotServiceAddGlobalWayPoint(const aubo_robot_namespace::wayPoint_S &wayPoint);
```

```
d.
e.   int     robotServiceAddGlobalWayPoint(const double jointAngle[aubo_robot_namespace::ARM_DOF]);
f.
g.   void    robotServiceGetGlobalWayPointVector(std::vector<aubo_robot_namespace::wayPoint_S> &wayPointVector);
```

轨迹运动需要的路点的数目不固定，因此轨迹运动的路点通过上面几个接口进行添加和清除。已知 6 个关节角，可以得到对应位置详细的路点信息。因此可以通过路点信息和关节角的形式来添加路点。对路点容器属性在轨迹运动的时候生效，即函数 robotServiceTrackMove（）。

设置和获取机械臂运动属性中的交融半径属性

```
a.   float robotServiceGetGlobalBlendRadius();
b.
c.   int     robotServiceSetGlobalBlendRadius(float value);
```

对交融半径的设置与获取，该属性仅在轨迹运动类型==CARTESIAN_MOVEP(MOVEP 轨迹)时，该属性生效，即函数 robotServiceTrackMove（），并且 subMoveMode==CARTESIAN_MOVEP.单位：米

设置和获取机械臂运动属性中的圆轨迹的圈数

```
a.   int     robotServiceGetGlobalCircularLoopTimes();
b.
c.   void    robotServiceSetGlobalCircularLoopTimes(int times);
```

当轨迹运动类型等于 ARC_CIR 时该属性生效；且圆轨迹的圈数等于零时，ARC_CIR 代表圆弧，圆轨迹的圈数大于零时，ARC_CIR 代表圆；

设置和获取机械臂运动属性中的偏移量属性

```
a.   int    robotServiceSetMoveRelativeParam(const aubo_robot_namespace::MoveRelative &relativeMoveOnBase);
b.
c.   int    robotServiceSetMoveRelativeParam(const aubo_robot_namespace::MoveRelative     &relativeMoveOnUserCoord,
                                             const aubo_robot_namespace::CoordCalibrateByJointAngleAndTool
d.   &userCoord);   //基于自定义坐标系
```

函数功能：设置运动属性中的偏移属性 单位米

设置机械臂运动属性中的示教坐标系

```
a.   int robotServiceSetTeachCoordinateSystem(const aubo_robot_namespace::CoordCalibrateByJointAngleAndTool
     &coordSystem);
```

函数功能：设置示教运动的参考坐标系。

参数描述：coordSystem 参考坐标系 具体参见该类型的解释;

返回值：调用成功返回 ErrnoSucc;错误返回错误号

关节运动(真正的运动)

```
a.  int robotServiceJointMove(double jointAngle[aubo_robot_namespace::ARM_DOF], bool IsBolck);
```

函数功能：机械臂通过关节运动的形式运动至目标路点。该运动属于关节型运动，因此在调用之前应该设置关节型运动的最大速度和最大加速度，如果用到偏移量的话需要设置偏移量属性。

参数描述：1、jointAngle 目标位置的关节角;

2、IsBolck 调用接口时是否阻塞;

返回值：调用成功返回 ErrnoSucc;错误返回错误号

直线运动

```
a.  int robotServiceLineMove(double jointAngle[aubo_robot_namespace::ARM_DOF],  bool IsBolck);
```

函数功能：机械臂通过直线运动的形式运动至目标路点。该运动属于末端型运动，因此在调用之前应该设置末端型运动的最大线速度，最大线加速度，最大角速度，最大角加速度；如果用到偏移量的话需要设置偏移量属性。

参数描述：1、jointAngle 目标位置的关节角;

2、IsBolck 调用接口时是否阻塞;

返回值：调用成功返回 ErrnoSucc;错误返回错误号

注意：如果用到了工具需要在运动之前设置工具的运动学参数（参见接口:robotServiceSetToolKinematicsParam）和动力学参数(参见接口：robotServiceSetToolDynamicsParam)。

旋转运动

```
a.  int robotServiceRotateMove(const aubo_robot_namespace::CoordCalibrateByJointAngleAndTool &userCoord, const double
    rotateAxisOnUserCoord[3], double rotateAngle,  bool IsBolck);
```

函数功能：机械臂的末端点绕着指定转轴做旋转运动（位置保持不变）；注意如果用到了工具需要在运动之前设置工具的运动学参数（参见接口:robotServiceSetToolKinematicsParam）和动力学参数(参见接口：robotServiceSetToolDynamicsParam)。

参数描述：1、userCoord 为转轴（rotateAxisOnUserCoord）的参考坐标系;

2、rotateAxisOnUserCoord 用户坐标系下自由矢量（可以理解为对应坐标系原点到同值坐

标点的矢量），用户坐标系标定通过第一个参数 userCoord 给出。

3、rotateAngle 绕转轴转动的转角，单位弧度。

4、IsBolck 调用接口时是否阻塞；

返回值：调用成功返回 ErrnoSucc;错误返回错误号

轨迹运动

```
a.  int robotServiceTrackMove(aubo_robot_namespace::move_track subMoveMode,      bool IsBolck);
```

函数功能：机械臂轨迹运动，根据 subMoveMode 类型的不同，该运动属于不同的运动类型

参数描述：

1 、 subMoveMode 当 subMoveMode==ARC_CIR, CARTESIAN_MOVEP, CARTESIAN_CUBICSPLINE, CARTESIAN_UBSPLINEINTP 时，该运动属于末端型运动；

当 subMoveMode==JIONT_CUBICSPLINE, JOINT_UBSPLINEINTP 时，该运动属于关节型运动；

当 subMoveMode==ARC_CIR 表示圆或者圆弧

当圆的圈数属性（CircularLoopTimes）为 0 时，表示圆弧轨迹，

当圆的圈数属性（CircularLoopTimes）大于 0 时，表示圆轨迹。

当 subMoveMode==CARTESIAN_MOVEP 表示 MOVEP 轨迹，需要用户这只交融半径的属性。

2、IsBolck 调用接口时是否阻塞；

返回值：调用成功返回 ErrnoSucc;错误返回错误号

注意：如果用到了工具需要在运动之前设置工具的运动学参数（参见接口:robotServiceSetToolKinematicsParam）和动力学参数(参见接口：robotServiceSetToolDynamicsParam)。

直线运动至目标位置

```
a.  int robotMoveLineToTargetPosition(const aubo_robot_namespace::CoordCalibrateByJointAngleAndTool   &userCoord,
b.                                    const aubo_robot_namespace::Pos                  &positionOnUserCoord,
c.                                    const aubo_robot_namespace::ToolInEndDesc    &toolInEndDesc,      //相对于用户坐标系的目标位置
d.                                    bool IsBolck = false);
```

函数功能：机械臂通过直线运动的形式运动至目标位置。该运动属于末端型运动，因此在调用之前应该设置末端型运动的最大线速度，最大线加速度，最大角速度，最大角加速度；如果用到偏移量的话需要设置偏移量属性。

参数描述：1、userCoord 目标位置（positionOnUserCoord）的参考坐标系;

2、positionOnUserCoord 工具末端点的位置信息（目标位置 x,y,z），工具参数通过下个参数（toolInEndDesc）给出;

3、toolInEndDesc 工具参数当没有使用工具时，将此参数设置为 0;

4、IsBolck 调用接口时是否阻塞；

返回值：调用成功返回 ErrnoSucc;错误返回错误号

注 意：如果用到了工具需要在运动之前设置工具的运动学参数（参见接口:robotServiceSetToolKinematicsParam）和动力学参数(参见接口：robotServiceSetToolDynamicsParam)。

通过关节运动的形式运动是目标位置

```
a.  int robotMoveLineToTargetPosition(const aubo_robot_namespace::CoordCalibrateByJointAngleAndTool  &userCoord,
b.                                    const aubo_robot_namespace::Pos                  &positionOnUserCoord,
                                      const aubo_robot_namespace::ToolInEndDesc    &toolInEndDesc,      //相对于用
c.  户坐标系的目标位置
d.                                    bool IsBolck = false);
```

参考 robotMoveLineToTargetPosition，只是运动的形式不同。因此在调用之前应该设置关节型运动的最大速度和最大加速度，如果用到偏移量的话需要设置偏移量属性。

示教运动的启动停止

```
a.  int robotServiceTeachStart(aubo_robot_namespace::teach_mode mode, bool direction);
```

函数功能：启动示教运动。

参数描述：1、mode 示教关节:JOINT1, JOINT2, JOINT3, JOINT4, JOINT5, JOINT6,

位置示教:MOV_X, MOV_Y, MOV_Z

姿态示教:ROT_X, ROT_Y, ROT_Z;

2、direction 方向正方向 true 反方向 false；

返回值：调用成功返回 ErrnoSucc;错误返回错误号

注 意：如果用到了工具需要在运动之前设置工具的运动学参数（参见接口:robotServiceSetToolKinematicsParam）和动力学参数(参见接口：robotServiceSetToolDynamicsParam)。

```
a.  int robotServiceTeachStop();
```

函数功能：停止当前的示教运动

返回值：调用成功返回 ErrnoSucc;错误返回错误号

运动的启动，停止，暂停，继续的控制

```
a.  int    rootServiceRobotMoveControl(aubo_robot_namespace::RobotMoveControlCommand cmd);
```

控制机械臂运动的启动，暂停，继续，停止。

离线轨迹运动

```
a.  int robotServiceOfflineTrackWaypointAppend(const std::vector<aubo_robot_namespace::wayPoint_S> &wayPointVector);
```

函数功能：通过路点容器追加离线轨迹路点到服务器
参数描述：wayPointVector 路点容器，容器最大允许 4000 个路点。
返回值：调用成功返回 ErrnoSucc;错误返回错误号

```
a.  int robotServiceOfflineTrackWaypointAppend(const char *fileName);
```

函数功能：通过路点文件的形式追加离线轨迹路点到服务器
参数描述：fileName 路点文件路径
返回值：调用成功返回 ErrnoSucc;错误返回错误号

```
a.  int robotServiceOfflineTrackWaypointClear ();
```

函数功能：清除服务器离线轨迹的路点
返回值：调用成功返回 ErrnoSucc;错误返回错误号

```
a.  int robotServiceOfflineTrackMoveStartup    (bool   IsBolck);
```

函数功能：启动离线轨迹的运行，该离线轨迹的路点的通过上面两个函数上传的。
参数描述：IsBolck 调用接口时是否阻塞；
返回值：调用成功返回 ErrnoSucc;错误返回错误号

```
a.  int robotServiceOfflineTrackMoveStop        ();
```

停止离线轨迹的运行。

3.5 API 之机械臂相关属性的获取与设置模块

机械臂当前工作模式的设置与获取

```
a.  int robotServiceGetRobotWorkMode(aubo_robot_namespace::RobotWorkMode &mode);
```

函数功能:获取当前机械臂工作模式
参数描述：mode　输出参数　仿真或真实的枚举类型
返回值：调用成功返回 ErrnoSucc;错误返回错误号

```
a.  int robotServiceSetRobotWorkMode(aubo_robot_namespace::RobotWorkMode mode);
```

函数功能:设置当前机械臂模式　仿真或真实
参数描述：mode 仿真或真实的枚举类型
返回值：调用成功返回 ErrnoSucc;错误返回错误号

判断真实机械臂是否存在

```
a.  int robotServiceGetIsRealRobotExist(bool &value);
```

函数功能:获取是否存在真实机械臂
参数描述：value　存在为 true,不存在为 false
返回值：调用成功返回 ErrnoSucc;错误返回错误号

获取机械臂关节状态

```
a.  int robotServiceGetRobotJointStatus(aubo_robot_namespace::JointStatus *jointStatus, int size);
```

函数功能:获取机械臂关节状态
参数描述：jointStatus 关节状态缓冲区　输出参数
size　缓冲区大小
返回值：调用成功返回 ErrnoSucc;错误返回错误号

获取机械臂诊断信息

```
a.  int robotServiceGetRobotDiagnosisInfo(aubo_robot_namespace::RobotDiagnosis &robotDiagnosisInfo);
```

函数功能:获取机械臂诊断信息
参数描述：robotDiagnosisInfo　机械臂诊断信息 输出参数
返回值：调用成功返回 ErrnoSucc;错误返回错误号

获取实时路点信息

```
a.  int robotServiceGetCurrentWaypointInfo(aubo_robot_namespace::wayPoint_S &wayPoint);
```

函数功能:获取机械臂实时路点信息
参数描述：wayPoint　机械臂当前路点信息
返回值：调用成功返回 ErrnoSucc;错误返回错误号

设置机械臂的碰撞等级

```
a.  int robotServiceSetRobotCollisionClass(int grade);
```

函数功能:设置碰撞等级
参数描述：grade　碰撞等级
返回值：调用成功返回 ErrnoSucc;错误返回错误号

3.6 API 之接口板 IO 模块

概述：

接口板 IO 主要分为两大块，分别是控制器 IO 和用户 IO，每个 IO 都有对应的名称和地址，在使用是可以通过名称或地址两种形式获取和设置 IO 的状态。

获取接口板 IO 配置信息

```
a.  int robotServiceGetBoardIOConfig(const std::vector<aubo_robot_namespace::RobotIoType> &ioType,
    std::vector<aubo_robot_namespace::RobotIoDesc> &configVector);
```

函数功能:获取接口板一个或多个 IO 类型的配置信息
参数描述：ioType　　IO 类型的容器　　输入参数
　　configVector　IO 配置信息集合　输出参数　配置信息中 RobotIoDesc.ioValue 是无效的
返回值：调用成功返回 ErrnoSucc;错误返回错误号

获取接口板 IO 的状态信息

```
1.  int robotServiceGetBoardIOStatus(const std::vector<aubo_robot_namespace::RobotIoType>  ioType,
    std::vector<aubo_robot_namespace::RobotIoDesc> &statusVector);
```

函数功能:获取接口板一个或多个 IO 类型的状态信息
参数描述：ioType　　IO 类型的容器　　输入参数
　　statusVector　IO 状态信息集合　输出参数
返回值：调用成功返回 ErrnoSucc;错误返回错误号

```
1.  int robotServiceGetBoardIOStatus(aubo_robot_namespace::RobotIoType type, std::string name, double &value);
```

函数功能:根据类型和名称获取指定接口板 IO 的状态信息
参数描述：type　IO 类型
　　name　IO 名称
　　value　IO 状态　输出参数
返回值：调用成功返回 ErrnoSucc;错误返回错误号

```
1.  int robotServiceGetBoardIOStatus(aubo_robot_namespace::RobotIoType type, int  addr,  double &value);
```

函数功能:根据类型和地址获取指定接口板 IO 的状态信息
参数描述：type　IO 类型
　　addr　IO 地址
　　value　IO 状态　输出参数
返回值：调用成功返回 ErrnoSucc;错误返回错误号

设置 IO 的状态

```
1.  int robotServiceSetBoardIOStatus(aubo_robot_namespace::RobotIoType type, std::string name, double value);
```

函数功能:根据类型和名称设置指定接口板 IO 的状态
参数描述：type　IO 类型
name　IO 名称
value　IO 状态
返回值：调用成功返回 ErrnoSucc;错误返回错误号

```
1.  int robotServiceSetBoardIOStatus(aubo_robot_namespace::RobotIoType type, int　addr, double value);
```

函数功能:根据类型和地址设置指定接口板 IO 的状态
参数描述：type　IO 类型
addr　IO 地址
value　IO 状态
返回值：调用成功返回 ErrnoSucc;错误返回错误号

查看联机模式状态

```
1.  int robotServiceIsOnlineMode(bool &isOnlineMode);
```

函数功能:当前机械臂是否运行在联机模式
参数描述：isOnlineMode　输出参数　返回 true 表示运行在联机模式否则表示非联机模式
返回值：调用成功返回 ErrnoSucc;错误返回错误号

```
1.  int robotServiceIsOnlineMasterMode(bool &isOnlineMasterMode);
```

函数功能:当前机械臂是否运行在联机主模式
参数描述：isOnlineMasterMode　输出参数　true：联机主模式 false：联机从模式
返回值：调用成功返回 ErrnoSucc;错误返回错误号

4 故障排除

错误号	错误信息
0	成功
10001	通用失败
10002	参数错误
10003	Socket 连接失败
10004	Socket 连接失败
10005	Socket 连接失败
10006	Socket 断开连接
10007	创建请求失败
10008	求相关的内部变量出错
10009	请求超时
10010	发送请求信息失败
10011	响应信息为空
10012	解析响应失败
10013	正解出错
10014	逆解出错
10015	工具标定出错
10016	工具标定参数有错
10017	坐标系标定失败
10018	坐标系转用户座标失败
10019	用户坐标系转基座标失败
10020	运动相关的内部变量出错
10021	运动请求失败
10022	生成运动请求失败
10023	运动被事件中断
10024	运动相关的路点容器的长度不符合规定
10025	服务器响应返回错误
10026	真实机械臂不存在，因为有些接口只有在真是机械臂存在的情况下才可以被调用