
Livox C++ API Reference v2.1.0

Livox

Mar 24, 2020

CONTENTS:

1	Basic Types and Functions	1
2	General Functions	11
2.1	Query Device Information	11
2.2	Receive Point Cloud Data	12
2.3	Set Coordinate System	13
2.4	Error Message From Device	13
2.5	Configure Static/Dynamic IP	15
2.6	Disconnect Device	17
2.7	Reboot Device	17
3	Livox Hub Functions	19
3.1	Query Connected LiDAR Unit Information	19
3.2	Configure LiDAR Unit Mode	20
3.3	Query LiDAR Unit Status	21
3.4	Sampling Control	22
3.5	Slot Power Control	22
3.6	Configure Livox Hub Extrinsic Parameters	23
3.7	Enable Hub Calculating Extrinsic Parameters	26
3.8	Enable or Disable The Rain/Fog Suppression	26
3.9	Turn On or Off Fan of LiDAR Unit	27
3.10	Config Point Cloud Return Mode of LiDAR Unit	30
3.11	Config IMU Push Frequency of LiDAR Unit	32
4	LiDAR Functions	35
4.1	Configure LiDAR Mode	35
4.2	Sample Control	35
4.3	Configure LiDAR Extrinsic Parameters	36
4.4	Enable and Disable the Rain/Fog Suppression	37
4.5	Turn On or Off LiDAR's Fan	38
4.6	Config LiDAR's Point Cloud Return Mode	39
4.7	Config LiDAR's IMU Push Frequency	40
	Index	41

BASIC TYPES AND FUNCTIONS

enum DeviceType

Device type.

Values:

kDeviceTypeHub = 0

Livox Hub.

kDeviceTypeLidarMid40 = 1

Mid-40.

kDeviceTypeLidarTele = 2

Tele.

kDeviceTypeLidarHorizon = 3

Horizon.

enum LidarState

Lidar state.

Values:

kLidarStateInit = 0

Initialization state.

kLidarStateNormal = 1

Normal work state.

kLidarStatePowerSaving = 2

Power-saving state.

kLidarStateStandBy = 3

Standby state.

kLidarStateError = 4

Error state.

kLidarStateUnknown = 5

Unknown state.

enum LidarFeature

Lidar feature.

Values:

kLidarFeatureNone = 0

No feature.

kLidarFeatureRainFog = 1

Rain and fog feature.

enum LidarIpMode

Lidar IP mode.

Values:

kLidarDynamicIpMode = 0
Dynamic IP.

kLidarStaticIpMode = 1
Static IP.

enum LivoxStatus

Function return value definition.

Values:

kStatusSendFailed = -9
Command send failed.

kStatusHandlerImplNotExist = -8
Handler implementation not exist.

kStatusInvalidHandle = -7
Device handle invalid.

kStatusChannelNotExist = -6
Command channel not exist.

kStatusNotEnoughMemory = -5
No enough memory.

kStatusTimeout = -4
Operation timeouts.

kStatusNotSupported = -3
Operation is not supported on this device.

kStatusNotConnected = -2
Requested device is not connected.

kStatusFailure = -1
Failure.

kStatusSuccess = 0
Success.

typedef int32_t livox_status

Function return value definition, refer to [LivoxStatus](#).

enum DeviceEvent

Device update type, indicating the change of device connection or working state.

Values:

kEventConnect = 0
Device is connected.

kEventDisconnect = 1
Device is removed.

kEventStateChange = 2
Device working state changes or an error occurs.

kEventHubConnectionChange = 3
Hub is connected or LiDAR unit(s) is/are removed.

enum TimestampType

Timestamp sync mode define.

Values:

kTimestampTypeNoSync = 0
No sync signal mode.

kTimestampTypePtp = 1
1588v2.0 PTP sync mode.

kTimestampTypeRsvd = 2
Reserved use.

kTimestampTypePpsGps = 3
pps+gps sync mode.

kTimestampTypePps = 4
pps only sync mode.

kTimestampTypeUnknown = 5
Unknown mode.

enum PointDataType
Point data type.

Values:

kCartesian
Cartesian coordinate point cloud.

kSpherical
Spherical coordinate point cloud.

kExtendCartesian
Extend cartesian coordinate point cloud.

kExtendSpherical
Extend spherical coordinate point cloud.

kDualExtendCartesian
Dual extend cartesian coordinate point cloud.

kDualExtendSpherical
Dual extend spherical coordinate point cloud.

kImu
IMU data.

kMaxPointDataType
Max Point Data Type.

enum PointCloudReturnMode
Point cloud return mode.

Values:

kFirstReturn
First single return mode .

kStrongestReturn
Strongest single return mode.

kDualReturn
Dual return mode.

enum ImuFreq
IMU push frequency.

Values:

kImuFreq0Hz
IMU push closed.

kImuFreq200Hz
IMU push frequency 200Hz.

struct LivoxRawPoint

Cartesian coordinate format.

Public Members

int32_t **x**

X axis, Unit:mm

int32_t **y**

Y axis, Unit:mm

int32_t **z**

Z axis, Unit:mm

uint8_t **reflectivity**

Reflectivity

struct LivoxSpherPoint

Spherical coordinate format.

Public Members

uint32_t **depth**

Depth, Unit: mm

uint16_t **theta**

Zenith angle[0, 18000], Unit: 0.01 degree

uint16_t **phi**

Azimuth[0, 36000], Unit: 0.01 degree

uint8_t **reflectivity**

Reflectivity

struct LivoxPoint

Standard point cloud format

Public Members

float **x**

X axis, Unit:m

float **y**

Y axis, Unit:m

float **z**

Z axis, Unit:m

uint8_t **reflectivity**

Reflectivity

struct LivoxExtendRawPoint

Extend cartesian coordinate format.

Public Members

int32_t **x**
X axis, Unit:mm

int32_t **y**
Y axis, Unit:mm

int32_t **z**
Z axis, Unit:mm

uint8_t **reflectivity**
Reflectivity

uint8_t **tag**
Tag

struct LivoxExtendSpherPoint
Extend spherical coordinate format.

Public Members

uint32_t **depth**
Depth, Unit: mm

uint16_t **theta**
Zenith angle[0, 18000], Unit: 0.01 degree

uint16_t **phi**
Azimuth[0, 36000], Unit: 0.01 degree

uint8_t **reflectivity**
Reflectivity

uint8_t **tag**
Tag

struct LivoxDualExtendRawPoint
Dual extend cartesian coordinate format.

Public Members

int32_t **x1**
X axis, Unit:mm

int32_t **y1**
Y axis, Unit:mm

int32_t **z1**
Z axis, Unit:mm

uint8_t **reflectivity1**
Reflectivity

uint8_t **tag1**
Tag

int32_t **x2**
X axis, Unit:mm

int32_t **y2**
Y axis, Unit:mm

int32_t **z2**
Z axis, Unit:mm

uint8_t **reflectivity2**
Reflectivity

uint8_t **tag2**
Tag

struct LivoxDualExtendSpherPoint
Dual extend spherical coordinate format.

Public Members

uint16_t **theta**
Zenith angle[0, 18000], Unit: 0.01 degree

uint16_t **phi**
Azimuth[0, 36000], Unit: 0.01 degree

uint32_t **depth1**
Depth, Unit: mm

uint8_t **reflectivity1**
Reflectivity

uint8_t **tag1**
Tag

uint32_t **depth2**
Depth, Unit: mm

uint8_t **reflectivity2**
Reflectivity

uint8_t **tag2**
Tag

struct LivoxImuPoint
IMU data format.

Public Members

float **gyro_x**
Gyroscope X axis, Unit:rad/s

float **gyro_y**
Gyroscope Y axis, Unit:rad/s

float **gyro_z**
Gyroscope Z axis, Unit:rad/s

float **acc_x**
Accelerometer X axis, Unit:g

float **acc_y**
Accelerometer Y axis, Unit:g

float **acc_z**
Accelerometer Z axis, Unit:g

struct DeviceInfo
Information of the connected LiDAR or hub.

Public Members

char **broadcast_code**[16]

Device broadcast code, null-terminated string, 15 characters at most.

uint8_t **handle**

Device handle.

uint8_t **slot**

Slot number used for connecting LiDAR.

uint8_t **id**

LiDAR id.

uint8_t **type**

Device type, refer to *DeviceType*.

uint16_t **data_port**

Point cloud data UDP port.

uint16_t **cmd_port**

Control command UDP port.

uint16_t **sensor_port**

IMU data UDP port.

char **ip**[16]

IP address.

LidarState **state**

LiDAR state.

LidarFeature **feature**

LiDAR feature.

StatusUnion **status**

LiDAR work state status.

uint8_t **firmware_version**[4]

Firmware version.

union StatusUnion

#include <livox_def.h> Information of LiDAR work state.

Public Members

uint32_t **progress**

LiDAR work state switching progress.

ErrorMessage **status_code**

LiDAR work state status code.

struct ReturnCode

Public Members

uint8_t **ret_code**
Return code.

char **broadcast_code**[16]
Device broadcast code.

struct LivoxSdkVersion
The numeric version information struct.

Public Members

int **major**
major number

int **minor**
minor number

int **patch**
patch number

void **GetLivoxSdkVersion** (*LivoxSdkVersion* *version)
Return SDK's version information in a numeric form.

Parameters

- **version**: Pointer to a version structure for returning the version information.

void **DisableConsoleLogger** ()
Disable console log output.

bool **Init** ()
Initialize the SDK.

Return true if successfully initialized, otherwise false.

bool **Start** ()
Start the device scanning routine which runs on a separate thread.

Return true if successfully started, otherwise false.

void **Uninit** ()
Uninitialize the SDK.

struct BroadcastDeviceInfo
The information of broadcast device.

Public Members

char **broadcast_code**[16]
Device broadcast code, null-terminated string, 15 characters at most.

uint8_t **dev_type**
Device type, refer to *DeviceType*.

uint16_t **reserved**
Reserved.

char **ip**[16]
Device ip.

typedef void (***DeviceBroadcastCallback**) (const *BroadcastDeviceInfo* *info)
SetBroadcastCallback response callback function.

Parameters

- `info`: information of the broadcast device, becomes invalid after the function returns.

void **SetBroadcastCallback** (*DeviceBroadcastCallback* cb)

Set the callback of listening device broadcast message. When broadcast message is received from Livox Hub/LiDAR, cb is called.

Parameters

- `cb`: callback for device broadcast.

typedef void (***DeviceStateUpdateCallback**) (const *DeviceInfo* *device, *DeviceEvent* type)

SetDeviceStateUpdateCallback response callback function.

Parameters

- `device`: information of the connected device.
- `type`: the update type that indicates connection/disconnection of the device or change of working state.

void **SetDeviceStateUpdateCallback** (*DeviceStateUpdateCallback* cb)

Add a callback for device connection or working state changing event.

Note Livox SDK supports two hardware connection modes. 1: Directly connecting to the LiDAR device; 2. Connecting to the LiDAR device(s) via the Livox Hub. In the first mode, connection/disconnection of every LiDAR unit is reported by this callback. In the second mode, only connection/disconnection of the Livox Hub is reported by this callback. If you want to get information of the LiDAR unit(s) connected to hub, see `HubQueryLidarInformation`.

Note 3 conditions can trigger this callback:

1. Connection and disconnection of device.
2. A change of device working state.
3. An error occurs.

Parameters

- `cb`: callback for device connection/disconnection.

livox_status **AddHubToConnect** (const char *broadcast_code, uint8_t *handle)

Add a broadcast code to the connecting list and only devices with broadcast code in this list will be connected. The broadcast code is unique for every device.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `broadcast_code`: device's broadcast code.
- `handle`: device handle. For Livox Hub, the handle is always 31; for LiDAR units connected to the Livox Hub, the corresponding handle is (slot-1)*3+id-1.

livox_status **AddLidarToConnect** (const char *broadcast_code, uint8_t *handle)

Add a broadcast code to the connecting list and only devices with broadcast code in this list will be connected. The broadcast code is unique for every device.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `broadcast_code`: device's broadcast code.
- `handle`: device handle. The handle is the same as the order calling `AddLidarToConnect` starting from 0.

livox_status **GetConnectedDevices** (*DeviceInfo* *devices, uint8_t *size)

Get all connected devices' information.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `devices`: list of connected devices' information.
- `size`: number of devices connected.

GENERAL FUNCTIONS

2.1 Query Device Information

struct DeviceInformationResponse

The response body of querying device information.

Public Members

uint8_t **ret_code**

Return code.

uint8_t **firmware_version**[4]

Firmware version.

typedef void (*DeviceInformationCallback) (*livox_status* status, uint8_t handle, *DeviceInformationResponse* *response, void *client_data)

Function type of callback that queries device's information.

Parameters

- **status**: kStatusSuccess on successful return, kStatusTimeout on timeout, see *LivoxStatus* for other error code.
- **handle**: device handle.
- **response**: response from the device.
- **client_data**: user data associated with the command.

livox_status **QueryDeviceInformation** (uint8_t *handle*, *DeviceInformationCallback* *cb*, void **client_data*)

Command to query device's information.

Return kStatusSuccess on successful return, see *LivoxStatus* for other error code.

Parameters

- **handle**: device handle.
- **cb**: callback for the command.
- **client_data**: user data associated with the command.

2.2 Receive Point Cloud Data

struct LivoxEthPacket

Point cloud packet.

Public Members

uint8_t **version**

Packet protocol version.

uint8_t **slot**

Slot number used for connecting LiDAR.

uint8_t **id**

LiDAR id.

uint8_t **rsvd**

Reserved.

uint32_t **err_code**

Device error status indicator information.

uint8_t **timestamp_type**

Timestamp type.

uint8_t **data_type**

Point cloud coordinate format, refer to *PointDataType* .

uint8_t **timestamp**[8]

Nanosecond or UTC format timestamp.

uint8_t **data**[1]

Point cloud data.

typedef void (*DataCallback) (uint8_t handle, *LivoxEthPacket* *data, uint32_t data_num, void *client_data)

Callback function for receiving point cloud data.

Parameters

- handle: device handle.
- data: device's data.
- data_num: number of points in data.
- client_data: user data associated with the command.

void **SetDataCallback** (uint8_t handle, *DataCallback* cb, void *client_data)

Set the callback to receive point cloud data. Only one callback is supported for a specific device. Set the point cloud data callback before beginning sampling.

Parameters

- handle: device handle.
- cb: callback to receive point cloud data.
- client_data: user data associated with the command.

livox_status **HubGetLidarHandle** (uint8_t slot, uint8_t id)

Get the LiDAR unit handle used in the Livox Hub data callback function from slot and id.

Return LiDAR unit handle.

Parameters

- slot: Livox Hub's slot.

- `id`: Livox Hub's id.

2.3 Set Coordinate System

livox_status **SetCartesianCoordinate** (uint8_t *handle*, *CommonCommandCallback* *cb*, void **client_data*)

Change point cloud coordinate system to cartesian coordinate.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `handle`: device handle.
- `cb`: callback for the command.
- `client_data`: user data associated with the command.

livox_status **SetSphericalCoordinate** (uint8_t *handle*, *CommonCommandCallback* *cb*, void **client_data*)

Change point cloud coordinate system to spherical coordinate.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `handle`: device handle.
- `cb`: callback for the command.
- `client_data`: user data associated with the command.

2.4 Error Message From Device

union ErrorMessage

#include <livox_def.h> Device error message.

Public Members

uint32_t **error_code**

Error code.

LidarErrorCode **lidar_error_code**

Lidar error code.

HubErrorCode **hub_error_code**

Hub error code.

struct LidarErrorCode

LiDAR error code.

Public Members

`uint32_t temp_status` : 2
0: Temperature in Normal State. 1: High or Low. 2: Extremely High or Extremely Low.

`uint32_t volt_status` : 2
0: Voltage in Normal State. 1: High. 2: Extremely High.

`uint32_t motor_status` : 2
0: Motor in Normal State. 1: Motor in Warning State. 2: Motor in Error State, Unable to Work.

`uint32_t dirty_warn` : 2
0: Not Dirty or Blocked. 1: Dirty or Blocked.

`uint32_t firmware_err` : 1
0: Firmware is OK. 1: Firmware is Abnormal, Need to be Upgraded.

`uint32_t pps_status` : 1
0: No PPS Signal. 1: PPS Signal is OK.

`uint32_t device_status` : 1
0: Normal. 1: Warning for Approaching the End of Service Life.

`uint32_t fan_status` : 1
0: Fan in Normal State. 1: Fan in Warning State.

`uint32_t self_heating` : 1
0: Normal. 1: Low Temperature Self Heating On.

`uint32_t ptp_status` : 1
0: No 1588 Signal. 1: 1588 Signal is OK.

`uint32_t time_sync_status` : 3
0: System dose not start time synchronization. 1: Using PTP 1588 synchronization. 2: Using GPS synchronization. 3: Using PPS synchronization. 4: System time synchronization is abnormal.(The highest priority synchronization signal is abnormal)

`uint32_t rsvd` : 13
Reserved.

`uint32_t system_status` : 2
0: Normal. 1: Warning. 2: Error.

struct HubErrorCode

Hub error code.

Public Members

`uint32_t sync_status` : 2
0: No synchronization signal. 1: 1588 synchronization. 2: GPS synchronization. 3: System time synchronization is abnormal.(The highest priority synchronization signal is abnormal)

`uint32_t temp_status` : 2
0: Temperature in Normal State. 1: High or Low. 2: Extremely High or Extremely Low.

`uint32_t lidar_status` : 1
0: LiDAR State is Normal. 1: LiDAR State is Abnormal.

`uint32_t lidar_link_status` : 1
0: LiDAR Connection is Normal. 1: LiDAR Connection is Abnormal.

`uint32_t firmware_err` : 1
0: LiDAR Firmware is OK. 1: LiDAR Firmware is Abnormal, Need to be Upgraded.

`uint32_t rsvd` : 23
Reserved.

uint32_t **system_status** : 2
 0: Normal. 1: Warning. 2: Error.

typedef void (*ErrorMessageCallback) (*livox_status* status, uint8_t handle, *ErrorMessage* *message)

Callback of the error status message. kStatusSuccess on successful return, see *LivoxStatus* for other

Parameters

- handle: device handle.
- response: response from the device.

livox_status **SetErrorMessageCallback** (uint8_t handle, *ErrorMessageCallback* cb)

Add error status callback for the device. error code.

Return kStatusSuccess on successful return, see *LivoxStatus* for other error code.

Parameters

- handle: device handle.
- cb: callback for the command.

2.5 Configure Static/Dynamic IP

struct SetDeviceIPModeRequest

The request body of the command for setting device's IP mode.

Public Members

uint8_t **ip_mode**

IP address mode: 0 for dynamic IP address, 1 for static IP address.

uint32_t **ip_addr**

IP address.

livox_status **SetStaticDynamicIP** (uint8_t handle, *SetDeviceIPModeRequest* *req, *CommonCommandCallback* cb, void *client_data)

Set device's IP mode.

Note *SetStaticDynamicIP* only supports setting Hub or Mid40/100's IP mode. If you want to set Horizon or Tele's IP mode, see *SetStaticIp* and *SetDynamicIp*.

Return kStatusSuccess on successful return, see *LivoxStatus* for other error code.

Parameters

- handle: device handle.
- req: request sent to device.
- cb: callback for the command.
- client_data: user data associated with the command.

struct SetStaticDeviceIpModeRequest

The request body of the command for setting static device's IP mode.

Public Members

uint32_t **ip_addr**

IP address.

uint32_t **net_mask**

Subnet mask.

uint32_t **gw_addr**

Gateway address.

livox_status **SetStaticIp** (uint8_t *handle*, *SetStaticDeviceIpModeRequest* **req*, *CommonCommandCallback* *cb*, void **client_data*)

Set device's static IP mode.

Note Mid40/100 is not supported to set subnet mask and gateway address. *SetStaticDeviceIpModeRequest*'s setting: *net_mask* and *gw_addr* will not take effect on Mid40/100.

Return *kStatusSuccess* on successful return, see *LivoxStatus* for other error code.

Parameters

- *handle*: device handle.
- *req*: request sent to device.
- *cb*: callback for the command.
- *client_data*: user data associated with the command.

livox_status **SetDynamicIp** (uint8_t *handle*, *CommonCommandCallback* *cb*, void **client_data*)

Set device's dynamic IP mode.

Return *kStatusSuccess* on successful return, see *LivoxStatus* for other error code.

Parameters

- *handle*: device handle.
- *cb*: callback for the command.
- *client_data*: user data associated with the command.

struct **GetDeviceIpModeResponse**

The response body of getting device's IP mode.

Public Members

uint8_t **ret_code**

Return code.

uint8_t **ip_mode**

IP address mode: 0 for dynamic IP address, 1 for static IP address.

uint32_t **ip_addr**

IP address.

uint32_t **net_mask**

Subnet mask.

uint32_t **gw_addr**

Gateway address.

typedef void (***GetDeviceIpInformationCallback**) (*livox_status* *status*, uint8_t *handle*, *GetDeviceIpModeResponse* **response*, void **client_data*)

Callback function that gets device's IP information.

Parameters

- **status:** `kStatusSuccess` on successful return, `kStatusTimeout` on timeout, see [LivoxStatus](#) for other error code.
- **handle:** device handle.
- **response:** response from the device.
- **client_data:** user data associated with the command.

livox_status **GetDeviceIpInformation** (uint8_t *handle*, *GetDeviceIpInformationCallback* *cb*, void **client_data*)

Get device's IP mode.

Return `kStatusSuccess` on successful return, see [LivoxStatus](#) for other error code.

Parameters

- **handle:** device handle.
- **cb:** callback for the command.
- **client_data:** user data associated with the command.

2.6 Disconnect Device

livox_status **DisconnectDevice** (uint8_t *handle*, *CommonCommandCallback* *cb*, void **client_data*)

Disconnect device.

Return `kStatusSuccess` on successful return, see [LivoxStatus](#) for other error code.

Parameters

- **handle:** device handle.
- **cb:** callback for the command.
- **client_data:** user data associated with the command.

2.7 Reboot Device

livox_status **RebootDevice** (uint8_t *handle*, uint16_t *timeout*, *CommonCommandCallback* *cb*, void **client_data*)

Reboot device.

Note *RebootDevice* is not supported for Mid40/100 firmware version < 03.07.0000.

Return `kStatusSuccess` on successful return, see [LivoxStatus](#) for other error code.

Parameters

- **handle:** device handle.
- **timeout:** reboot device after [timeout] ms.
- **cb:** callback for the command.
- **client_data:** user data associated with the command.

LIVOX HUB FUNCTIONS

3.1 Query Connected LiDAR Unit Information

struct ConnectedLidarInfo

The information of LiDAR units that are connected to the Livox Hub.

Public Members

char **broadcast_code**[16]

Device broadcast code, null-terminated string, 15 characters at most.

uint8_t **dev_type**

Device type, refer to *DeviceType*.

uint8_t **version**[4]

Firmware version.

uint8_t **slot**

Slot number used for connecting LiDAR units.

uint8_t **id**

Device id.

struct HubQueryLidarInformationResponse

The response body of querying the information of LiDAR units connected to the Livox Hub.

Public Members

uint8_t **ret_code**

Return code.

uint8_t **count**

Count of device_info_list.

ConnectedLidarInfo **device_info_list**[1]

Connected lidars information list.

typedef void (***HubQueryLidarInformationCallback**)(*livox_status* status, uint8_t handle,
HubQueryLidarInformationResponse
*response, void *client_data)

HubQueryLidarInformation response callback function.

Parameters

- status: kStatusSuccess on successful return, kStatusTimeout on timeout, see *LivoxStatus* for other error code.
- handle: device handle.
- response: response from the device.

- `client_data`: user data associated with the command.

3.2 Configure LiDAR Unit Mode

struct HubSetModeResponse

The response of setting Livox Hub's working mode.

Public Members

`uint8_t ret_code`

Return code.

`uint8_t count`

Count of `ret_state_list`.

ReturnCode `ret_state_list[1]`

Return status list.

typedef void (***HubSetModeCallback**) (*livox_status* status, uint8_t handle, *HubSetModeResponse* *response, void *client_data)

HubSetMode response callback function.

Parameters

- `status`: `kStatusSuccess` on successful return, `kStatusTimeout` on timeout, see *LivoxStatus* for other error code.
- `handle`: device handle.
- `response`: response from the device.
- `client_data`: user data associated with the command.

struct HubSetModeRequest

The request body of setting Livox Hub's working mode.

Public Members

`uint8_t count`

Count of `config_list`.

LidarModeRequestItem `config_list[1]`

LiDAR mode configuration list.

struct LidarModeRequestItem

LiDAR mode configuration information.

Public Members

char `broadcast_code[16]`

Device broadcast code, null-terminated string, 15 characters at most.

`uint8_t state`

LiDAR state, refer to *LidarMode*.

livox_status **HubSetMode** (*HubSetModeRequest* *req, uint16_t length, *HubSetModeCallback* cb, void *client_data)

Set the mode of LiDAR unit connected to the Livox Hub.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `req`: mode configuration of LiDAR units.
- `length`: length of `req`.
- `cb`: callback for the command.
- `client_data`: user data associated with the command.

3.3 Query LiDAR Unit Status

struct LidarStateItem

Public Members

char **broadcast_code**[16]
Broadcast code.

uint8_t **state**
LiDAR state.

uint8_t **feature**
LiDAR feature.

StatusUnion **error_union**
LiDAR work state.

struct HubQueryLidarStatusResponse

The response body of getting sub LiDAR's state conncted to Hub.

Public Members

uint8_t **ret_code**
Return code.

uint8_t **count**
Count of `state_list`.

LidarStateItem **state_list**[1]
LiDAR units state list.

typedef void (*HubQueryLidarStatusCallback) (*livox_status* status, uint8_t handle, *HubQueryLidarStatusResponse* *response, void *client_data)

HubQueryLidarStatus response callback function.

Parameters

- `status`: `kStatusSuccess` on successful return, `kStatusTimeout` on timeout, see *LivoxStatus* for other error code.
- `handle`: device handle.
- `response`: response from the device.
- `client_data`: user data associated with the command.

livox_status **HubQueryLidarStatus** (*HubQueryLidarStatusCallback* cb, void *client_data)

Get the state of LiDAR units connected to the Livox Hub.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `cb`: callback for the command.

- `client_data`: user data associated with the command.

3.4 Sampling Control

typedef void (***CommonCommandCallback**) (*livox_status* status, uint8_t handle, uint8_t response, void *client_data)

Function type of callback with 1 byte of response.

Parameters

- `status`: `kStatusSuccess` on successful return, `kStatusTimeout` on timeout, see *LivoxStatus* for other error code.
- `handle`: device handle.
- `response`: response from the device.
- `client_data`: user data associated with the command.

livox_status **HubStartSampling** (*CommonCommandCallback* cb, void *client_data)
Start hub sampling.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `cb`: callback for the command.
- `client_data`: user data associated with the command.

livox_status **HubStopSampling** (*CommonCommandCallback* cb, void *client_data)
Stop the Livox Hub's sampling.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `cb`: callback for the command.
- `client_data`: user data associated with the command.

3.5 Slot Power Control

struct **HubControlSlotPowerRequest**

The request body of toggling the power supply of the slot.

Public Members

uint8_t **slot**
Slot of the hub.

uint8_t **state**
Status of toggling the power supply.

livox_status **HubControlSlotPower** (*HubControlSlotPowerRequest* *req, *CommonCommandCallback* cb, void *client_data)

Toggle the power supply of designated slots.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `req`: request whether to enable or disable the power of designated slots.
- `cb`: callback for the command.

- `client_data`: user data associated with the command.

struct HubQuerySlotPowerStatusResponse

The response body of getting Hub slots' power state.

Public Members

`uint8_t ret_code`

Return code.

`uint16_t slot_power_state`

Slot power status.

typedef void (***HubQuerySlotPowerStatusCallback**)(*livox_status* status, uint8_t handle, *HubQuerySlotPowerStatusResponse* *response, void *client_data)

HubQuerySlotPowerStatus response callback function.

Parameters

- `status`: `kStatusSuccess` on successful return, `kStatusTimeout` on timeout, see *LivoxStatus* for other error code.
- `handle`: device handle.
- `response`: response from the device.
- `client_data`: user data associated with the command.

livox_status **HubQuerySlotPowerStatus** (*HubQuerySlotPowerStatusCallback* cb, void *client_data)

Get the power supply state of each hub slot.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `cb`: callback for the command.
- `client_data`: user data associated with the command.

3.6 Configure Livox Hub Extrinsic Parameters

struct HubSetExtrinsicParameterResponse

The response body of setting the Livox Hub's parameters.

Public Members

`uint8_t ret_code`

Return code.

`uint8_t count`

Count of `ret_code_list`.

ReturnCode `ret_code_list[1]`

Return code list.

typedef void (***HubSetExtrinsicParameterCallback**)(*livox_status* status, uint8_t handle, *HubSetExtrinsicParameterResponse* *response, void *client_data)

HubSetExtrinsicParameter response callback function.

Parameters

- **status**: `kStatusSuccess` on successful return, `kStatusTimeout` on timeout, see [LivoxStatus](#) for other error code.
- **handle**: device handle.
- **response**: response from the device.
- **client_data**: user data associated with the command.

struct HubSetExtrinsicParameterRequest

The request body of setting the Livox Hub's parameters.

Public Members

`uint8_t count`

Count of `cfg_param_list`.

[ExtrinsicParameterRequestItem](#) **parameter_list**[1]

Extrinsic parameter configuration list.

struct ExtrinsicParameterRequestItem

LiDAR configuration information.

Public Members

`char broadcast_code`[16]

Device broadcast code.

`float roll`

Roll angle, unit: degree.

`float pitch`

Pitch angle, unit: degree.

`float yaw`

Yaw angle, unit: degree.

`int32_t x`

X translation, unit: mm.

`int32_t y`

Y translation, unit: mm.

`int32_t z`

Z translation, unit: mm.

livox_status **HubSetExtrinsicParameter** (*HubSetExtrinsicParameterRequest* *req, `uint16_t` length, *HubSetExtrinsicParameterCallback* cb, void *client_data)

Set extrinsic parameters of LiDAR units connected to the Livox Hub.

Return `kStatusSuccess` on successful return, see [LivoxStatus](#) for other error code.

Parameters

- **req**: the parameters to write.
- **length**: the request's length.
- **cb**: callback for the command.
- **client_data**: user data associated with the command.

struct HubGetExtrinsicParameterRequest

The request body of getting the Livox Hub's parameters.

Public Members**uint8_t count**

Count of code_list.

DeviceBroadcastCode **code_list[1]**

Broadcast code list.

struct DeviceBroadcastCode**Public Members**char **broadcast_code[16]**

Device broadcast code.

struct HubGetExtrinsicParameterResponse

The response body of getting the Livox Hub's parameters.

Public Members**uint8_t ret_code**

Return code.

uint8_t count

Count of code_list.

ExtrinsicParameterResponseItem **parameter_list[1]**

Extrinsic parameter list.

struct ExtrinsicParameterResponseItem

LiDAR extrinsic parameters.

Public Members**uint8_t ret_code**

Return code.

char **broadcast_code[16]**

Broadcast code.

float **roll**

Roll angle, unit: degree.

float **pitch**

Pitch angle, unit: degree.

float **yaw**

Yaw angle, unit: degree.

int32_t **x**

X translation, unit: mm.

int32_t **y**

Y translation, unit: mm.

int32_t **z**

Z translation, unit: mm.

typedef void (*HubGetExtrinsicParameterCallback) (*livox_status* status, uint8_t handle,
HubGetExtrinsicParameterResponse
 *response, void *client_data)

HubGetExtrinsicParameter response callback function.

Parameters

- `status`: `kStatusSuccess` on successful return, `kStatusTimeout` on timeout, see [LivoxStatus](#) for other error code.
- `handle`: device handle.
- `response`: response from the device.
- `client_data`: user data associated with the command.

livox_status **HubGetExtrinsicParameter** (*HubGetExtrinsicParameterCallback* *cb*, *void* **client_data*)

Get extrinsic parameters of LiDAR units connected to the Livox Hub.

Return `kStatusSuccess` on successful return, see [LivoxStatus](#) for other error code.

Parameters

- `cb`: callback for the command.
- `client_data`: user data associated with the command.

3.7 Enable Hub Calculating Extrinsic Parameters

livox_status **HubExtrinsicParameterCalculation** (*bool enable*, *CommonCommandCallback* *cb*, *void* **client_data*)

Turn on or off the calculation of extrinsic parameters.

Return `kStatusSuccess` on successful return, see [LivoxStatus](#) for other error code.

Parameters

- `enable`: the request whether enable or disable calculating the extrinsic parameters.
- `cb`: callback for the command.
- `client_data`: user data associated with the command.

3.8 Enable or Disable The Rain/Fog Suppression

struct **RainFogSuppressRequestItem**

Public Members

char **broadcast_code**[16]
Device broadcast code.

uint8_t **feature**
Close or open the rain and fog feature.

struct **HubRainFogSuppressRequest**

The request body of toggling the Livox Hub's rain and fog mode.

Public Members

`uint8_t count`

Count of `lidar_cfg_list`.

RainFogSuppressRequestItem `lidar_cfg_list[1]`

Rain fog suppress configuration list.

struct `HubRainFogSuppressResponse`

The response body of toggling the Livox Hub's rain and fog mode.

Public Members

`uint8_t ret_code`

Return code.

`uint8_t count`

Count of `ret_state_list`.

ReturnCode `ret_state_list[1]`

Return state list

typedef `void (*HubRainFogSuppressCallback) (livox_status status, uint8_t handle, HubRainFogSuppressResponse *response, void *client_data)`

`HubRainFogSuppress` response callback function.

Parameters

- `status`: `kStatusSuccess` on successful return, `kStatusTimeout` on timeout, see *LivoxStatus* for other error code.
- `handle`: device handle.
- `response`: response from the device.
- `client_data`: user data associated with the command.

livox_status **HubRainFogSuppress** (*HubRainFogSuppressRequest* *req, `uint16_t length`, *HubRainFogSuppressCallback* cb, `void *client_data`)

Toggling the rain and fog mode for lidars connected to the hub.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `req`: the request whether open or close the rain and fog mode.
- `length`: the request's length.
- `cb`: callback for the command.
- `client_data`: user data associated with the command.

3.9 Turn On or Off Fan of LiDAR Unit

struct `FanControlRequestItem`

Public Members

char **broadcast_code**[16]
Device broadcast code.

uint8_t **state**
Fan state: 1 for turn on fan, 0 for turn off fan.

struct HubFanControlRequest
The request body of controlling the sub LiDAR's fan state conneted to Hub.

Public Members

uint8_t **count**
Count of lidar_cfg_list.

FanControlRequestItem **lidar_cfg_list**[1]
Fan control configuration list.

struct HubFanControlResponse
The response body of controlling the sub LiDAR's fan state conneted to Hub.

Public Members

uint8_t **ret_code**
Return code.

uint8_t **count**
Count of return_list.

ReturnCode **return_list**[1]
Return list

typedef void (*HubFanControlCallback) (*livox_status* status, uint8_t handle, *HubFanControlResponse* *response, void *client_data)
HubFanControl response callback function.

Parameters

- status: kStatusSuccess on successful return, kStatusTimeout on timeout, see *LivoxStatus* for other error code.
- handle: device handle.
- response: response from the device.
- client_data: user data associated with the command.

livox_status **HubFanControl** (*HubFanControlRequest* *req, uint16_t length, *HubFanControlCallback* cb, void *client_data)
Turn on or off the fan of LiDAR unit connected to the Livox Hub.

Return kStatusSuccess on successful return, see *LivoxStatus* for other error code.

Parameters

- req: Fan control of LiDAR units.
- length: length of req.
- cb: callback for the command.
- client_data: user data associated with the command.

struct GetFanStateRequestItem

Public Members

char **broadcast_code**[16]
Device broadcast code.

struct HubGetFanStateRequest

The request body of getting the sub LiDAR's fan state conneted to Hub.

Public Members

uint8_t **count**
Count of lidar_cfg_list.

GetFanStateRequestItem **lidar_cfg_list**[1]
Get Fan state list.

struct GetFanStateResponseItem

Public Members

uint8_t **ret_code**
Return code.

char **broadcast_code**[16]
Device broadcast code.

uint8_t **state**
Fan state: 1 for fan is on, 0 for fan is off.

struct HubGetFanStateResponse

The response body of getting the sub LiDAR's fan state conneted to Hub.

Public Members

uint8_t **ret_code**
Return code.

uint8_t **count**
Count of return_list.

GetFanStateResponseItem **return_list**[1]
Fan state list.

typedef void (*HubGetFanStateCallback) (*livox_status* status, uint8_t handle, *HubGetFanStateResponse* *response, void *client_data)

HubGetFanControl response callback function.

Parameters

- status: kStatusSuccess on successful return, kStatusTimeout on timeout, see *LivoxStatus* for other error code.
- handle: device handle.
- response: response from the device.
- client_data: user data associated with the command.

livox_status **HubGetFanState** (*HubGetFanStateRequest* *req, uint16_t length, *HubGetFanStateCallback* cb, void *client_data)

Get fan state of LiDAR unit connected to the Livox Hub.

Return kStatusSuccess on successful return, see *LivoxStatus* for other error code.

Parameters

- req: Get fan state of LiDAR units.
- length: length of req.
- cb: callback for the command.
- client_data: user data associated with the command.

3.10 Config Point Cloud Return Mode of LiDAR Unit

struct SetPointCloudReturnModeRequestItem

Public Members

char **broadcast_code**[16]
Device broadcast code.

uint8_t **mode**
Point cloud return mode, refer to [PointCloudReturnMode](#).

struct HubSetPointCloudReturnModeRequest

The request body of setting point cloud return mode of sub LiDAR conneted to Hub.

Public Members

uint8_t **count**
Count of lidar_cfg_list.

[SetPointCloudReturnModeRequestItem](#) **lidar_cfg_list**[1]
Point cloud return mode configuration list.

struct HubSetPointCloudReturnModeResponse

The response body of setting point cloud return mode of sub LiDAR conneted to Hub.

Public Members

uint8_t **ret_code**
Return code.

uint8_t **count**
Count of return_list.

[ReturnCode](#) **return_list**[1]
Return list.

typedef void (*HubSetPointCloudReturnModeCallback) ([livox_status](#) status, uint8_t handle,
[HubSetPointCloudReturnModeResponse](#) *response, void
*client_data)

HubSetPointCloudReturnMode response callback function.

Parameters

- status: kStatusSuccess on successful return, kStatusTimeout on timeout, see [LivoxStatus](#) for other error code.
- handle: device handle.
- response: response from the device.
- client_data: user data associated with the command.

livox_status **HubSetPointCloudReturnMode** (*HubSetPointCloudReturnModeRequest* *req, uint16_t length, *HubSetPointCloudReturnModeCallback* cb, void *client_data)

Set point cloud return mode of LiDAR units connected to the Livox Hub.

Return kStatusSuccess on successful return, see *LivoxStatus* for other error code.

Parameters

- req: set point cloud return mode of LiDAR units.
- length: the request's length.
- cb: callback for the command.
- client_data: user data associated with the command.

struct **GetPointCloudReturnModeRequestItem**

Public Members

char **broadcast_code**[16]
Device broadcast code.

struct **HubGetPointCloudReturnModeRequest**

The request body of getting sub LiDAR's point cloud return mode conneted to Hub.

Public Members

uint8_t **count**
Count of lidar_cfg_list.

GetPointCloudReturnModeRequestItem **lidar_cfg_list**[1]
Get point cloud return mode list.

struct **GetPointCloudReturnModeResponseItem**

Public Members

uint8_t **ret_code**
Return code.

char **broadcast_code**[16]
Device broadcast code.

uint8_t **mode**
Point cloud return mode, refer to *PointCloudReturnMode*.

struct **HubGetPointCloudReturnModeResponse**

The response body of getting sub LiDAR's point cloud return mode conneted to Hub.

Public Members

uint8_t **ret_code**
Return code.

uint8_t **count**
Count of return_list.

GetPointCloudReturnModeResponseItem **return_list**[1]
Point cloud return mode list.

```
typedef void (*HubGetPointCloudReturnModeCallback) (livox_status status, uint8_t handle,
                                                    HubGetPointCloudReturnModeResponse *response, void
                                                    *client_data)
```

HubGetPointCloudReturnMode response callback function.

Parameters

- status: kStatusSuccess on successful return, kStatusTimeout on timeout, see [LivoxStatus](#) for other error code.
- handle: device handle.
- response: response from the device.
- client_data: user data associated with the command.

```
livox_status HubGetPointCloudReturnMode (HubGetPointCloudReturnModeRequest *req,
                                         uint16_t length, HubGetPointCloudReturnMode-
                                         Callback cb, void *client_data)
```

Get point cloud return mode of LiDAR unit connected to the Livox Hub.

Return kStatusSuccess on successful return, see [LivoxStatus](#) for other error code.

Parameters

- req: Get point cloud return mode of LiDAR units.
- length: length of req.
- cb: callback for the command.
- client_data: user data associated with the command.

3.11 Config IMU Push Frequency of LiDAR Unit

```
struct SetImuPushFrequencyRequestItem
```

Public Members

char **broadcast_code**[16]
Device broadcast code.

uint8_t **freq**
IMU push frequency, refer to [ImuFreq](#).

```
struct HubSetImuPushFrequencyRequest
```

The request body of setting IMU push frequency of sub LiDAR conneted to Hub.

Public Members

uint8_t **count**
Count of lidar_cfg_list.

[SetImuPushFrequencyRequestItem](#) **lidar_cfg_list**[1]
IMU push frequency configuration list.

```
struct HubSetImuPushFrequencyResponse
```

The response body of setting IMU push frequency of sub LiDAR conneted to Hub.

Public Members

uint8_t **ret_code**

Return code.

uint8_t **count**

Count of return_list.

ReturnCode **return_list**[1]

Return list.

```
typedef void (*HubSetImuPushFrequencyCallback) (livox_status status, uint8_t handle,
                                                HubSetImuPushFrequencyResponse
                                                *response, void *client_data)
```

HubSetImuPushFrequency response callback function.

Parameters

- status: kStatusSuccess on successful return, kStatusTimeout on timeout, see *LivoxStatus* for other error code.
- handle: device handle.
- response: response from the device.
- client_data: user data associated with the command.

```
livox_status HubSetImuPushFrequency (HubSetImuPushFrequencyRequest *req, uint16_t length,
                                     HubSetImuPushFrequencyCallback cb, void *client_data)
```

Set IMU push frequency of LiDAR units connected to the Livox Hub.

Return kStatusSuccess on successful return, see *LivoxStatus* for other error code.

Parameters

- req: set IMU push frequency of LiDAR units.
- length: the request's length.
- cb: callback for the command.
- client_data: user data associated with the command.

```
struct GetImuPushFrequencyRequestItem
```

Public Members

char **broadcast_code**[16]

Device broadcast code.

```
struct HubGetImuPushFrequencyRequest
```

The request body of getting sub LiDAR's IMU push frequency conneted to Hub.

Public Members

uint8_t **count**

Count of lidar_cfg_list.

GetImuPushFrequencyRequestItem **lidar_cfg_list**[1]

Get IMU push frequency list.

```
struct GetImuPushFrequencyResponseItem
```

Public Members

uint8_t **ret_code**

Return code.

char **broadcast_code**[16]

Device broadcast code.

uint8_t **freq**

IMU push frequency, refer to *ImuFreq*.

struct HubGetImuPushFrequencyResponse

The response body of getting sub LiDAR's IMU push frequency conneted to Hub.

Public Members

uint8_t **ret_code**

Return code.

uint8_t **count**

Count of return_list.

GetImuPushFrequencyResponseItem **return_list**[1]

IMU push frequency list.

typedef void (*HubGetImuPushFrequencyCallback) (*livox_status* status, uint8_t handle,
HubGetImuPushFrequencyResponse
*response, void *client_data)

HubGetImuPushFrequency response callback function.

Parameters

- status: kStatusSuccess on successful return, kStatusTimeout on timeout, see *LivoxStatus* for other error code.
- handle: device handle.
- response: response from the device.
- client_data: user data associated with the command.

livox_status **HubGetImuPushFrequency** (*HubGetImuPushFrequencyRequest* *req, uint16_t length,
HubGetImuPushFrequencyCallback cb, void *client_data)

Get IMU push frequency of LiDAR units connected to the Livox Hub.

Return kStatusSuccess on successful return, see *LivoxStatus* for other error code.

Parameters

- req: get IMU push frequency of LiDAR units.
- length: the request's length.
- cb: callback for the command.
- client_data: user data associated with the command.

LIDAR FUNCTIONS

4.1 Configure LiDAR Mode

enum LidarMode

Lidar mode.

Values:

kLidarModeNormal = 1

Normal mode.

kLidarModePowerSaving = 2

Power-saving mode.

kLidarModeStandby = 3

Standby mode.

livox_status **LidarSetMode** (uint8_t *handle*, *LidarMode* *mode*, *CommonCommandCallback* *cb*, void
**client_data*)

Set LiDAR mode.

Note Successful callback function status only means LiDAR successfully starting the changing process of mode. You need to observe the actually change of mode in DeviceStateUpdateCallback function.

Return kStatusSuccess on successful return, see *LivoxStatus* for other error code.

Parameters

- *handle*: device handle.
- *mode*: the mode to change.
- *cb*: callback for the command.
- *client_data*: user data associated with the command.

4.2 Sample Control

livox_status **LidarStartSampling** (uint8_t *handle*, *CommonCommandCallback* *cb*, void
**client_data*)

Start LiDAR sampling.

Return kStatusSuccess on successful return, see *LivoxStatus* for other error code.

Parameters

- *handle*: device handle.
- *cb*: callback for the command.
- *client_data*: user data associated with the command.

livox_status **LidarStopSampling** (uint8_t *handle*, *CommonCommandCallback* *cb*, void **client_data*)
Stop LiDAR sampling.

Return kStatusSuccess on successful return, see *LivoxStatus* for other error code.

Parameters

- *handle*: device handle.
- *cb*: callback for the command.
- *client_data*: user data associated with the command.

4.3 Configure LiDAR Extrinsic Parameters

struct LidarSetExtrinsicParameterRequest

The request body for the command of setting Livox LiDAR's parameters.

Public Members

float **roll**
Roll angle, unit: degree.

float **pitch**
Pitch angle, unit: degree.

float **yaw**
Yaw angle, unit: degree.

int32_t **x**
X translation, unit: mm.

int32_t **y**
Y translation, unit: mm.

int32_t **z**
Z translation, unit: mm.

livox_status **LidarSetExtrinsicParameter** (uint8_t *handle*, *LidarSetExtrinsicParameterRequest* **req*, *CommonCommandCallback* *cb*, void **client_data*)

Set LiDAR extrinsic parameters.

Return kStatusSuccess on successful return, see *LivoxStatus* for other error code.

Parameters

- *handle*: device handle.
- *req*: the parameters to write.
- *cb*: callback for the command.
- *client_data*: user data associated with the command.

struct LidarGetExtrinsicParameterResponse

The response body of getting Livox LiDAR's parameters.

Public Members

uint8_t **ret_code**

float **roll**

Roll angle, unit: degree.

float **pitch**

Pitch angle, unit: degree.

float **yaw**

Yaw angle, unit: degree.

int32_t **x**

X translation, unit: mm.

int32_t **y**

Y translation, unit: mm.

int32_t **z**

Z translation, unit: mm.

```
typedef void (*LidarGetExtrinsicParameterCallback) (livox_status status, uint8_t
                                                    handle, LidarGetExtrinsicParam-
                                                    eterResponse *response, void
                                                    *client_data)
```

LidarGetExtrinsicParameter response callback function.

Parameters

- **status**: kStatusSuccess on successful return, kStatusTimeout on timeout, see [LivoxStatus](#) for other error code.
- **handle**: device handle.
- **response**: response from the device.
- **client_data**: user data associated with the command.

```
livox_status LidarGetExtrinsicParameter (uint8_t handle, LidarGetExtrinsicParameterCallback
                                          cb, void *client_data)
```

Get LiDAR extrinsic parameters.

Return kStatusSuccess on successful return, see [LivoxStatus](#) for other error code.

Parameters

- **handle**: device handle.
- **cb**: callback for the command.
- **client_data**: user data associated with the command.

4.4 Enable and Disable the Rain/Fog Suppression

```
livox_status LidarRainFogSuppress (uint8_t handle, bool enable, CommonCommandCallback cb,
                                   void *client_data)
```

Enable and disable the rain/fog suppression.

Note [LidarRainFogSuppress](#) only support for Mid40/100.

Return kStatusSuccess on successful return, see [LivoxStatus](#) for other error code.

Parameters

- **handle**: device handle.
- **enable**: enable and disable the rain/fog suppression.

- `cb`: callback for the command.
- `client_data`: user data associated with the command.

4.5 Turn On or Off LiDAR's Fan

livox_status **LidarTurnOnFan** (uint8_t *handle*, *CommonCommandCallback* *cb*, void **client_data*)
Turn on the fan.

Note *LidarTurnOnFan* is not supported for Mid40/100.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `handle`: device handle.
- `cb`: callback for the command.
- `client_data`: user data associated with the command.

livox_status **LidarTurnOffFan** (uint8_t *handle*, *CommonCommandCallback* *cb*, void **client_data*)
Turn off the fan.

Note *LidarTurnOffFan* is not supported for Mid40/100.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `handle`: device handle.
- `cb`: callback for the command.
- `client_data`: user data associated with the command.

struct LidarGetFanStateResponse
The response body of getting the Livox LiDAR's fan state.

Public Members

uint8_t **ret_code**
Return code.

uint8_t **state**
Fan state: 1 for fan is on, 0 for fan is off.

typedef void (*LidarGetFanStateCallback) (*livox_status* *status*, uint8_t *handle*, *LidarGetFanStateResponse* **response*, void **client_data*)
LidarGetFanState response callback function.

Parameters

- `status`: `kStatusSuccess` on successful return, `kStatusTimeout` on timeout, see *LivoxStatus* for other error code.
- `handle`: device handle.
- `response`: response from the device.
- `client_data`: user data associated with the command.

livox_status **LidarGetFanState** (uint8_t *handle*, *LidarGetFanStateCallback* *cb*, void **client_data*)
Get state of the fan.

Note *LidarGetFanState* is not supported for Mid40/100.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `handle`: device handle.
- `cb`: callback for the command.
- `client_data`: user data associated with the command.

4.6 Config LiDAR's Point Cloud Return Mode

livox_status **LidarSetPointCloudReturnMode** (uint8_t *handle*, *PointCloudReturnMode* *mode*, *CommonCommandCallback* *cb*, void **client_data*)

Set point cloud return mode.

Note *LidarSetPointCloudReturnMode* is not supported for Mid40/100.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `handle`: device handle.
- `mode`: point cloud return mode.
- `cb`: callback for the command.
- `client_data`: user data associated with the command.

struct LidarGetPointCloudReturnModeResponse

The response body of getting the Livox LiDAR's point cloud return mode.

Public Members

uint8_t **ret_code**
Return code.

uint8_t **mode**
Point cloud return mode, refer to *PointCloudReturnMode*.

typedef void (***LidarGetPointCloudReturnModeCallback**) (*livox_status* *status*, uint8_t *handle*, *LidarGetPointCloudReturnModeResponse* **response*, void **client_data*)

LidarGetPointCloudReturnMode response callback function.

Parameters

- `status`: `kStatusSuccess` on successful return, `kStatusTimeout` on timeout, see *LivoxStatus* for other error code.
- `handle`: device handle.
- `response`: response from the device.
- `client_data`: user data associated with the command.

livox_status **LidarGetPointCloudReturnMode** (uint8_t *handle*, *LidarGetPointCloudReturnModeCallback* *cb*, void **client_data*)

Get point cloud return mode.

Note *LidarGetPointCloudReturnMode* is not supported for Mid40/100.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `handle`: device handle.

- `cb`: callback for the command.
- `client_data`: user data associated with the command.

4.7 Config LiDAR's IMU Push Frequency

livox_status **LidarSetImuPushFrequency** (uint8_t *handle*, *ImuFreq* *freq*, *CommonCommandCallback* *cb*, void **client_data*)

Set IMU push frequency.

Note *LidarSetImuPushFrequency* is not supported for Mid40/100.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `handle`: device handle.
- `freq`: IMU push frequency.
- `cb`: callback for the command.
- `client_data`: user data associated with the command.

struct **LidarGetImuPushFrequencyResponse**

The response body of getting the Livox LiDAR's IMU push frequency.

Public Members

uint8_t **ret_code**

Return code.

uint8_t **freq**

IMU push frequency, refer to *ImuFreq*.

typedef void (***LidarGetImuPushFrequencyCallback**) (*livox_status* *status*, uint8_t *handle*, *LidarGetImuPushFrequencyResponse* **response*, void **client_data*)

LidarGetImuPushFrequency response callback function.

Parameters

- `status`: `kStatusSuccess` on successful return, `kStatusTimeout` on timeout, see *LivoxStatus* for other error code.
- `handle`: device handle.
- `response`: response from the device.
- `client_data`: user data associated with the command.

livox_status **LidarGetImuPushFrequency** (uint8_t *handle*, *LidarGetImuPushFrequencyCallback* *cb*, void **client_data*)

Get IMU push frequency.

Note *LidarGetImuPushFrequency* is not supported for Mid40/100.

Return `kStatusSuccess` on successful return, see *LivoxStatus* for other error code.

Parameters

- `handle`: device handle.
- `cb`: callback for the command.
- `client_data`: user data associated with the command.

A

AddHubToConnect (C++ *function*), 9
AddLidarToConnect (C++ *function*), 9

B

BroadcastDeviceInfo (C++ *class*), 8
BroadcastDeviceInfo
 ::broadcast_code (C++ *member*), 8
BroadcastDeviceInfo
 ::dev_type (C++ *member*), 8
BroadcastDeviceInfo
 ::ip (C++ *member*), 8
BroadcastDeviceInfo
 ::reserved (C++ *member*), 8

C

CommonCommandCallback (C++ *type*), 22
ConnectedLidarInfo (C++ *class*), 19
ConnectedLidarInfo
 ::broadcast_code (C++ *member*), 19
ConnectedLidarInfo
 ::dev_type (C++ *member*), 19
ConnectedLidarInfo
 ::id (C++ *member*), 19
ConnectedLidarInfo
 ::slot (C++ *member*), 19
ConnectedLidarInfo
 ::version (C++ *member*), 19

D

DataCallback (C++ *type*), 12
DeviceBroadcastCallback (C++ *type*), 8
DeviceBroadcastCode (C++ *class*), 25
DeviceBroadcastCode
 ::broadcast_code (C++ *member*), 25
DeviceEvent (C++ *enum*), 2
DeviceInfo (C++ *class*), 6
DeviceInfo
 ::broadcast_code (C++ *member*), 7
DeviceInfo
 ::cmd_port (C++ *member*), 7

DeviceInfo
 ::data_port (C++ *member*), 7
DeviceInfo
 ::feature (C++ *member*), 7
DeviceInfo
 ::firmware_version (C++ *member*), 7
DeviceInfo
 ::handle (C++ *member*), 7
DeviceInfo
 ::id (C++ *member*), 7
DeviceInfo
 ::ip (C++ *member*), 7
DeviceInfo
 ::sensor_port (C++ *member*), 7
DeviceInfo
 ::slot (C++ *member*), 7
DeviceInfo
 ::state (C++ *member*), 7
DeviceInfo
 ::status (C++ *member*), 7
DeviceInfo
 ::type (C++ *member*), 7
DeviceInformationCallback (C++ *type*), 11
DeviceInformationResponse (C++ *class*), 11
DeviceInformationResponse
 ::firmware_version (C++ *member*), 11
DeviceInformationResponse
 ::ret_code (C++ *member*), 11
DeviceStateUpdateCallback (C++ *type*), 9
DeviceType (C++ *enum*), 1
DisableConsoleLogger (C++ *function*), 8
DisconnectDevice (C++ *function*), 17

E

ErrorMessage (C++ *union*), 13
ErrorMessage
 ::error_code (C++ *member*), 13
ErrorMessage
 ::hub_error_code (C++ *member*), 13
ErrorMessage
 ::lidar_error_code (C++ *member*), 13
ErrorMessageCallback (C++ *type*), 15

ExtrinsicParameterRequestItem	(C++	GetDeviceIpModeResponse
<i>class</i>), 24		<i>::net_mask (C++ member)</i> , 16
ExtrinsicParameterRequestItem		GetDeviceIpModeResponse
<i>::broadcast_code (C++ member)</i> ,		<i>::ret_code (C++ member)</i> , 16
24		GetFanStateRequestItem (C++ <i>class</i>), 28
ExtrinsicParameterRequestItem		GetFanStateRequestItem
<i>::pitch (C++ member)</i> , 24		<i>::broadcast_code (C++ member)</i> ,
ExtrinsicParameterRequestItem		29
<i>::roll (C++ member)</i> , 24		GetFanStateResponseItem (C++ <i>class</i>), 29
ExtrinsicParameterRequestItem		GetFanStateResponseItem
<i>::x (C++ member)</i> , 24		<i>::broadcast_code (C++ member)</i> ,
ExtrinsicParameterRequestItem		29
<i>::y (C++ member)</i> , 24		GetFanStateResponseItem
ExtrinsicParameterRequestItem		<i>::ret_code (C++ member)</i> , 29
<i>::yaw (C++ member)</i> , 24		GetFanStateResponseItem
ExtrinsicParameterRequestItem		<i>::state (C++ member)</i> , 29
<i>::z (C++ member)</i> , 24		GetImuPushFrequencyRequestItem (C++
ExtrinsicParameterResponseItem (C++		<i>class</i>), 33
<i>class</i>), 25		GetImuPushFrequencyRequestItem
ExtrinsicParameterResponseItem		<i>::broadcast_code (C++ member)</i> ,
<i>::broadcast_code (C++ member)</i> ,		33
25		GetImuPushFrequencyResponseItem (C++
ExtrinsicParameterResponseItem		<i>class</i>), 33
<i>::pitch (C++ member)</i> , 25		GetImuPushFrequencyResponseItem
ExtrinsicParameterResponseItem		<i>::broadcast_code (C++ member)</i> ,
<i>::ret_code (C++ member)</i> , 25		34
ExtrinsicParameterResponseItem		GetImuPushFrequencyResponseItem
<i>::roll (C++ member)</i> , 25		<i>::freq (C++ member)</i> , 34
ExtrinsicParameterResponseItem		GetImuPushFrequencyResponseItem
<i>::x (C++ member)</i> , 25		<i>::ret_code (C++ member)</i> , 34
ExtrinsicParameterResponseItem		GetLivoxSdkVersion (C++ <i>function</i>), 8
<i>::y (C++ member)</i> , 25		GetPointCloudReturnModeRequestItem
ExtrinsicParameterResponseItem		(C++ <i>class</i>), 31
<i>::yaw (C++ member)</i> , 25		GetPointCloudReturnModeRequestItem
ExtrinsicParameterResponseItem		<i>::broadcast_code (C++ member)</i> , 31
<i>::z (C++ member)</i> , 25		GetPointCloudReturnModeResponseItem
		(C++ <i>class</i>), 31
		GetPointCloudReturnModeResponseItem
		<i>::broadcast_code (C++ member)</i> , 31
		GetPointCloudReturnModeResponseItem
		<i>::mode (C++ member)</i> , 31
		GetPointCloudReturnModeResponseItem
		<i>::ret_code (C++ member)</i> , 31
F		H
FanControlRequestItem (C++ <i>class</i>), 27		HubControlSlotPower (C++ <i>function</i>), 22
FanControlRequestItem		HubControlSlotPowerRequest (C++ <i>class</i>), 22
<i>::broadcast_code (C++ member)</i> ,		HubControlSlotPowerRequest
28		<i>::slot (C++ member)</i> , 22
FanControlRequestItem		HubControlSlotPowerRequest
<i>::state (C++ member)</i> , 28		<i>::state (C++ member)</i> , 22
		HubErrorCode (C++ <i>class</i>), 14
G		HubErrorCode
GetConnectedDevices (C++ <i>function</i>), 9		<i>::firmware_err (C++ member)</i> ,
GetDeviceIpInformation (C++ <i>function</i>), 17		14
GetDeviceIpInformationCallback (C++		HubErrorCode
<i>type</i>), 16		<i>::lidar_link_status (C++ member)</i> ,
GetDeviceIpModeResponse (C++ <i>class</i>), 16		14
GetDeviceIpModeResponse		
<i>::gw_addr (C++ member)</i> , 16		
GetDeviceIpModeResponse		
<i>::ip_addr (C++ member)</i> , 16		
GetDeviceIpModeResponse		
<i>::ip_mode (C++ member)</i> , 16		

HubErrorCode	HubGetFanStateResponse
::lidar_status (C++ member), 14	::ret_code (C++ member), 29
HubErrorCode	HubGetFanStateResponse
::rsvd (C++ member), 14	::return_list (C++ member), 29
HubErrorCode	HubGetImuPushFrequency (C++ function), 34
::sync_status (C++ member), 14	HubGetImuPushFrequencyCallback (C++ type), 34
HubErrorCode	HubGetImuPushFrequencyRequest (C++ class), 33
::system_status (C++ member), 14	HubGetImuPushFrequencyRequest
HubErrorCode	::count (C++ member), 33
::temp_status (C++ member), 14	HubGetImuPushFrequencyRequest
HubExtrinsicParameterCalculation (C++ function), 26	::lidar_cfg_list (C++ member), 33
HubFanControl (C++ function), 28	HubGetImuPushFrequencyResponse (C++ class), 34
HubFanControlCallback (C++ type), 28	HubGetImuPushFrequencyResponse
HubFanControlRequest (C++ class), 28	::count (C++ member), 34
HubFanControlRequest	HubGetImuPushFrequencyResponse
::count (C++ member), 28	::ret_code (C++ member), 34
HubFanControlRequest	HubGetImuPushFrequencyResponse
::lidar_cfg_list (C++ member), 28	::return_list (C++ member), 34
HubFanControlResponse (C++ class), 28	HubGetLidarHandle (C++ function), 12
HubFanControlResponse	HubGetPointCloudReturnMode (C++ function), 32
::count (C++ member), 28	HubGetPointCloudReturnModeCallback (C++ type), 31
HubFanControlResponse	HubGetPointCloudReturnModeRequest (C++ class), 31
::ret_code (C++ member), 28	HubGetPointCloudReturnModeRequest
HubFanControlResponse	::count (C++ member), 31
::return_list (C++ member), 28	HubGetPointCloudReturnModeRequest
HubGetExtrinsicParameter (C++ function), 26	::lidar_cfg_list (C++ member), 31
HubGetExtrinsicParameterCallback (C++ type), 25	HubGetPointCloudReturnModeResponse (C++ class), 31
HubGetExtrinsicParameterRequest (C++ class), 24	HubGetPointCloudReturnModeResponse
HubGetExtrinsicParameterRequest	::count (C++ member), 31
::code_list (C++ member), 25	HubGetPointCloudReturnModeResponse
HubGetExtrinsicParameterRequest	::ret_code (C++ member), 31
::count (C++ member), 25	HubGetPointCloudReturnModeResponse
HubGetExtrinsicParameterResponse (C++ class), 25	::return_list (C++ member), 31
HubGetExtrinsicParameterResponse	HubQueryLidarInformationCallback (C++ type), 19
::count (C++ member), 25	HubQueryLidarInformationResponse (C++ class), 19
HubGetExtrinsicParameterResponse	HubQueryLidarInformationResponse
::parameter_list (C++ member), 25	::count (C++ member), 19
HubGetExtrinsicParameterResponse	HubQueryLidarInformationResponse
::ret_code (C++ member), 25	::device_info_list (C++ member), 19
HubGetFanState (C++ function), 29	HubQueryLidarInformationResponse
HubGetFanStateCallback (C++ type), 29	::ret_code (C++ member), 19
HubGetFanStateRequest (C++ class), 29	HubQueryLidarStatus (C++ function), 21
HubGetFanStateRequest	HubQueryLidarStatusCallback (C++ type), 21
::count (C++ member), 29	HubQueryLidarStatusResponse (C++ class), 21
HubGetFanStateRequest	HubQueryLidarStatusResponse
::lidar_cfg_list (C++ member), 29	
HubGetFanStateResponse (C++ class), 29	
HubGetFanStateResponse	
::count (C++ member), 29	


```

        ::count (C++ member), 21
HubQueryLidarStatusResponse
        ::ret_code (C++ member), 21
HubQueryLidarStatusResponse
        ::state_list (C++ member), 21
HubQuerySlotPowerStatus (C++ function), 23
HubQuerySlotPowerStatusCallback (C++
    type), 23
HubQuerySlotPowerStatusResponse (C++
    class), 23
HubQuerySlotPowerStatusResponse
        ::ret_code (C++ member), 23
HubQuerySlotPowerStatusResponse
        ::slot_power_state (C++ mem-
            ber), 23
HubRainFogSuppress (C++ function), 27
HubRainFogSuppressCallback (C++ type), 27
HubRainFogSuppressRequest (C++ class), 26
HubRainFogSuppressRequest
        ::count (C++ member), 27
HubRainFogSuppressRequest
        ::lidar_cfg_list (C++ member),
            27
HubRainFogSuppressResponse (C++ class), 27
HubRainFogSuppressResponse
        ::count (C++ member), 27
HubRainFogSuppressResponse
        ::ret_code (C++ member), 27
HubRainFogSuppressResponse
        ::ret_state_list (C++ member),
            27
HubSetExtrinsicParameter (C++ function),
    24
HubSetExtrinsicParameterCallback (C++
    type), 23
HubSetExtrinsicParameterRequest (C++
    class), 24
HubSetExtrinsicParameterRequest
        ::count (C++ member), 24
HubSetExtrinsicParameterRequest
        ::parameter_list (C++ member),
            24
HubSetExtrinsicParameterResponse (C++
    class), 23
HubSetExtrinsicParameterResponse
        ::count (C++ member), 23
HubSetExtrinsicParameterResponse
        ::ret_code (C++ member), 23
HubSetExtrinsicParameterResponse
        ::ret_code_list (C++ member), 23
HubSetImuPushFrequency (C++ function), 33
HubSetImuPushFrequencyCallback (C++
    type), 33
HubSetImuPushFrequencyRequest (C++
    class), 32
HubSetImuPushFrequencyRequest
        ::count (C++ member), 32
HubSetImuPushFrequencyRequest
        ::lidar_cfg_list (C++ member),
            32
HubSetImuPushFrequencyResponse (C++
    class), 32
HubSetImuPushFrequencyResponse
        ::count (C++ member), 33
HubSetImuPushFrequencyResponse
        ::ret_code (C++ member), 33
HubSetImuPushFrequencyResponse
        ::return_list (C++ member), 33
HubSetMode (C++ function), 20
HubSetModeCallback (C++ type), 20
HubSetModeRequest (C++ class), 20
HubSetModeRequest
        ::config_list (C++ member), 20
HubSetModeRequest
        ::count (C++ member), 20
HubSetModeResponse (C++ class), 20
HubSetModeResponse
        ::count (C++ member), 20
HubSetModeResponse
        ::ret_code (C++ member), 20
HubSetModeResponse
        ::ret_state_list (C++ member),
            20
HubSetPointCloudReturnMode (C++ func-
            tion), 30
HubSetPointCloudReturnModeCallback
    (C++ type), 30
HubSetPointCloudReturnModeRequest
    (C++ class), 30
HubSetPointCloudReturnModeRequest
        ::count (C++ member), 30
HubSetPointCloudReturnModeRequest
        ::lidar_cfg_list (C++ member), 30
HubSetPointCloudReturnModeResponse
    (C++ class), 30
HubSetPointCloudReturnModeResponse
        ::count (C++ member), 30
HubSetPointCloudReturnModeResponse
        ::ret_code (C++ member), 30
HubSetPointCloudReturnModeResponse
        ::return_list (C++ member), 30
HubStartSampling (C++ function), 22
HubStopSampling (C++ function), 22

I
ImuFreq (C++ enum), 3
Init (C++ function), 8

K
kCartesian (C++ enumerator), 3
kDeviceTypeHub (C++ enumerator), 1
kDeviceTypeLidarHorizon (C++ enumerator),
    1
kDeviceTypeLidarMid40 (C++ enumerator), 1
kDeviceTypeLidarTele (C++ enumerator), 1
kDualExtendCartesian (C++ enumerator), 3

```


kDualExtendSpherical (C++ enumerator), 3
 kDualReturn (C++ enumerator), 3
 kEventConnect (C++ enumerator), 2
 kEventDisconnect (C++ enumerator), 2
 kEventHubConnectionChange (C++ enumerator), 2
 kEventStateChange (C++ enumerator), 2
 kExtendCartesian (C++ enumerator), 3
 kExtendSpherical (C++ enumerator), 3
 kFirstReturn (C++ enumerator), 3
 kImu (C++ enumerator), 3
 kImuFreq0Hz (C++ enumerator), 3
 kImuFreq200Hz (C++ enumerator), 3
 kLidarDynamicIpMode (C++ enumerator), 1
 kLidarFeatureNone (C++ enumerator), 1
 kLidarFeatureRainFog (C++ enumerator), 1
 kLidarModeNormal (C++ enumerator), 35
 kLidarModePowerSaving (C++ enumerator), 35
 kLidarModeStandby (C++ enumerator), 35
 kLidarStateError (C++ enumerator), 1
 kLidarStateInit (C++ enumerator), 1
 kLidarStateNormal (C++ enumerator), 1
 kLidarStatePowerSaving (C++ enumerator), 1
 kLidarStateStandBy (C++ enumerator), 1
 kLidarStateUnknown (C++ enumerator), 1
 kLidarStaticIpMode (C++ enumerator), 2
 kMaxPointDataType (C++ enumerator), 3
 kSpherical (C++ enumerator), 3
 kStatusChannelNotExist (C++ enumerator), 2
 kStatusFailure (C++ enumerator), 2
 kStatusHandlerImplNotExist (C++ enumerator), 2
 kStatusInvalidHandle (C++ enumerator), 2
 kStatusNotConnected (C++ enumerator), 2
 kStatusNotEnoughMemory (C++ enumerator), 2
 kStatusNotSupported (C++ enumerator), 2
 kStatusSendFailed (C++ enumerator), 2
 kStatusSuccess (C++ enumerator), 2
 kStatusTimeout (C++ enumerator), 2
 kStrongestReturn (C++ enumerator), 3
 kTimestampTypeNoSync (C++ enumerator), 2
 kTimestampTypePps (C++ enumerator), 3
 kTimestampTypePpsGps (C++ enumerator), 3
 kTimestampTypePtp (C++ enumerator), 2
 kTimestampTypeRsvd (C++ enumerator), 3
 kTimestampTypeUnknown (C++ enumerator), 3

L

LidarErrorCode (C++ class), 13
 LidarErrorCode
 ::device_status (C++ member), 14
 LidarErrorCode
 ::dirty_warn (C++ member), 14
 LidarErrorCode
 ::fan_status (C++ member), 14
 LidarErrorCode
 ::firmware_err (C++ member), 14
 LidarErrorCode
 ::motor_status (C++ member), 14
 LidarErrorCode
 ::pps_status (C++ member), 14
 LidarErrorCode
 ::ptp_status (C++ member), 14
 LidarErrorCode
 ::rsvd (C++ member), 14
 LidarErrorCode
 ::self_heating (C++ member), 14
 LidarErrorCode
 ::system_status (C++ member), 14
 LidarErrorCode
 ::temp_status (C++ member), 14
 LidarErrorCode
 ::time_sync_status (C++ member), 14
 LidarErrorCode
 ::volt_status (C++ member), 14
 LidarFeature (C++ enum), 1
 LidarGetExtrinsicParameter (C++ function), 37
 LidarGetExtrinsicParameterCallback (C++ type), 37
 LidarGetExtrinsicParameterResponse (C++ class), 36
 LidarGetExtrinsicParameterResponse
 ::pitch (C++ member), 37
 LidarGetExtrinsicParameterResponse
 ::ret_code (C++ member), 37
 LidarGetExtrinsicParameterResponse
 ::roll (C++ member), 37
 LidarGetExtrinsicParameterResponse
 ::x (C++ member), 37
 LidarGetExtrinsicParameterResponse
 ::y (C++ member), 37
 LidarGetExtrinsicParameterResponse
 ::yaw (C++ member), 37
 LidarGetExtrinsicParameterResponse
 ::z (C++ member), 37
 LidarGetFanState (C++ function), 38
 LidarGetFanStateCallback (C++ type), 38
 LidarGetFanStateResponse (C++ class), 38
 LidarGetFanStateResponse
 ::ret_code (C++ member), 38
 LidarGetFanStateResponse
 ::state (C++ member), 38
 LidarGetImuPushFrequency (C++ function), 40
 LidarGetImuPushFrequencyCallback (C++ type), 40
 LidarGetImuPushFrequencyResponse (C++ class), 40
 LidarGetImuPushFrequencyResponse

```

        ::freq (C++ member), 40
LidarGetImuPushFrequencyResponse
        ::ret_code (C++ member), 40
LidarGetPointCloudReturnMode (C++ func-
tion), 39
LidarGetPointCloudReturnModeCallback
(C++ type), 39
LidarGetPointCloudReturnModeResponse
(C++ class), 39
LidarGetPointCloudReturnModeResponse
::mode (C++ member), 39
LidarGetPointCloudReturnModeResponse
::ret_code (C++ member), 39
LidarIpMode (C++ enum), 1
LidarMode (C++ enum), 35
LidarModeRequestItem (C++ class), 20
LidarModeRequestItem
::broadcast_code (C++ member),
20
LidarModeRequestItem
::state (C++ member), 20
LidarRainFogSuppress (C++ function), 37
LidarSetExtrinsicParameter (C++ func-
tion), 36
LidarSetExtrinsicParameterRequest
(C++ class), 36
LidarSetExtrinsicParameterRequest
::pitch (C++ member), 36
LidarSetExtrinsicParameterRequest
::roll (C++ member), 36
LidarSetExtrinsicParameterRequest
::x (C++ member), 36
LidarSetExtrinsicParameterRequest
::y (C++ member), 36
LidarSetExtrinsicParameterRequest
::yaw (C++ member), 36
LidarSetExtrinsicParameterRequest
::z (C++ member), 36
LidarSetImuPushFrequency (C++ function),
40
LidarSetMode (C++ function), 35
LidarSetPointCloudReturnMode (C++ func-
tion), 39
LidarStartSampling (C++ function), 35
LidarState (C++ enum), 1
LidarStateItem (C++ class), 21
LidarStateItem
::broadcast_code (C++ member),
21
LidarStateItem
::error_union (C++ member), 21
LidarStateItem
::feature (C++ member), 21
LidarStateItem
::state (C++ member), 21
LidarStopSampling (C++ function), 35
LidarTurnOffFan (C++ function), 38
LidarTurnOnFan (C++ function), 38

livox_status (C++ type), 2
LivoxDualExtendRawPoint (C++ class), 5
LivoxDualExtendRawPoint
::reflectivity1 (C++ member),
5
LivoxDualExtendRawPoint
::reflectivity2 (C++ member),
5
LivoxDualExtendRawPoint
::tag1 (C++ member), 5
LivoxDualExtendRawPoint
::tag2 (C++ member), 6
LivoxDualExtendRawPoint
::x1 (C++ member), 5
LivoxDualExtendRawPoint
::x2 (C++ member), 5
LivoxDualExtendRawPoint
::y1 (C++ member), 5
LivoxDualExtendRawPoint
::y2 (C++ member), 5
LivoxDualExtendRawPoint
::z1 (C++ member), 5
LivoxDualExtendRawPoint
::z2 (C++ member), 5
LivoxDualExtendSpherPoint (C++ class), 6
LivoxDualExtendSpherPoint
::depth1 (C++ member), 6
LivoxDualExtendSpherPoint
::depth2 (C++ member), 6
LivoxDualExtendSpherPoint
::phi (C++ member), 6
LivoxDualExtendSpherPoint
::reflectivity1 (C++ member),
6
LivoxDualExtendSpherPoint
::reflectivity2 (C++ member),
6
LivoxDualExtendSpherPoint
::tag1 (C++ member), 6
LivoxDualExtendSpherPoint
::tag2 (C++ member), 6
LivoxDualExtendSpherPoint
::theta (C++ member), 6
LivoxEthPacket (C++ class), 12
LivoxEthPacket
::data (C++ member), 12
LivoxEthPacket
::data_type (C++ member), 12
LivoxEthPacket
::err_code (C++ member), 12
LivoxEthPacket
::id (C++ member), 12
LivoxEthPacket
::rsvd (C++ member), 12
LivoxEthPacket
::slot (C++ member), 12
LivoxEthPacket
::timestamp (C++ member), 12

```

LivoxEthPacket
 ::timestamp_type (C++ member), 12
 LivoxEthPacket
 ::version (C++ member), 12
 LivoxExtendRawPoint (C++ class), 4
 LivoxExtendRawPoint
 ::reflectivity (C++ member), 5
 LivoxExtendRawPoint
 ::tag (C++ member), 5
 LivoxExtendRawPoint
 ::x (C++ member), 5
 LivoxExtendRawPoint
 ::y (C++ member), 5
 LivoxExtendRawPoint
 ::z (C++ member), 5
 LivoxExtendSpherPoint (C++ class), 5
 LivoxExtendSpherPoint
 ::depth (C++ member), 5
 LivoxExtendSpherPoint
 ::phi (C++ member), 5
 LivoxExtendSpherPoint
 ::reflectivity (C++ member), 5
 LivoxExtendSpherPoint
 ::tag (C++ member), 5
 LivoxExtendSpherPoint
 ::theta (C++ member), 5
 LivoxImuPoint (C++ class), 6
 LivoxImuPoint
 ::acc_x (C++ member), 6
 LivoxImuPoint
 ::acc_y (C++ member), 6
 LivoxImuPoint
 ::acc_z (C++ member), 6
 LivoxImuPoint
 ::gyro_x (C++ member), 6
 LivoxImuPoint
 ::gyro_y (C++ member), 6
 LivoxImuPoint
 ::gyro_z (C++ member), 6
 LivoxPoint (C++ class), 4
 LivoxPoint
 ::reflectivity (C++ member), 4
 LivoxPoint
 ::x (C++ member), 4
 LivoxPoint
 ::y (C++ member), 4
 LivoxPoint
 ::z (C++ member), 4
 LivoxRawPoint (C++ class), 3
 LivoxRawPoint
 ::reflectivity (C++ member), 4
 LivoxRawPoint
 ::x (C++ member), 4
 LivoxRawPoint
 ::y (C++ member), 4
 LivoxRawPoint
 ::z (C++ member), 4
 LivoxSdkVersion (C++ class), 8
 LivoxSdkVersion
 ::major (C++ member), 8
 LivoxSdkVersion
 ::minor (C++ member), 8
 LivoxSdkVersion
 ::patch (C++ member), 8
 LivoxSpherPoint (C++ class), 4
 LivoxSpherPoint
 ::depth (C++ member), 4
 LivoxSpherPoint
 ::phi (C++ member), 4
 LivoxSpherPoint
 ::reflectivity (C++ member), 4
 LivoxSpherPoint
 ::theta (C++ member), 4
 LivoxStatus (C++ enum), 2

P

PointCloudReturnMode (C++ enum), 3
 PointDataType (C++ enum), 3

Q

QueryDeviceInformation (C++ function), 11

R

RainFogSuppressRequestItem (C++ class), 26
 RainFogSuppressRequestItem
 ::broadcast_code (C++ member), 26
 RainFogSuppressRequestItem
 ::feature (C++ member), 26
 RebootDevice (C++ function), 17
 ReturnCode (C++ class), 7
 ReturnCode
 ::broadcast_code (C++ member), 8
 ReturnCode
 ::ret_code (C++ member), 8

S

SetBroadcastCallback (C++ function), 9
 SetCartesianCoordinate (C++ function), 13
 SetDataCallback (C++ function), 12
 SetDeviceIPModeRequest (C++ class), 15
 SetDeviceIPModeRequest
 ::ip_addr (C++ member), 15
 SetDeviceIPModeRequest
 ::ip_mode (C++ member), 15
 SetDeviceStateUpdateCallback (C++ function), 9
 SetDynamicIp (C++ function), 16
 SetErrorMessageCallback (C++ function), 15

SetImuPushFrequencyRequestItem (C++
class), 32

SetImuPushFrequencyRequestItem
::broadcast_code (C++ member),
32

SetImuPushFrequencyRequestItem
::freq (C++ member), 32

SetPointCloudReturnModeRequestItem
(C++ class), 30

SetPointCloudReturnModeRequestItem
::broadcast_code (C++ member), 30

SetPointCloudReturnModeRequestItem
::mode (C++ member), 30

SetSphericalCoordinate (C++ function), 13

SetStaticDeviceIpModeRequest (C++ class),
15

SetStaticDeviceIpModeRequest
::gw_addr (C++ member), 16

SetStaticDeviceIpModeRequest
::ip_addr (C++ member), 16

SetStaticDeviceIpModeRequest
::net_mask (C++ member), 16

SetStaticDynamicIP (C++ function), 15

SetStaticIp (C++ function), 16

Start (C++ function), 8

StatusUnion (C++ union), 7

StatusUnion
::progress (C++ member), 7

StatusUnion
::status_code (C++ member), 7

T

TimestampType (C++ enum), 2

U

Uninit (C++ function), 8