数据广播接收 SDK

（VDMocapSDK_DataRead）说明

摘要

数据广播接收 SDK 包含头文件及动态链接库。其中头文件："VDMocapSDK_DataRead.h"，含 SDK 成员函数信息，通过这些成员函数可以获取软件广播数据；"VDMocapSDK_DataRead_DataType.h"，含成员函数用到的枚举类型及结构体信息。其中文件夹"Windows SDK"下的动态链接库支持 64 位，文件夹"Linux SDK"下的动态链接库支持 Ubuntu 20.04 和 22.04，系统架构支持 x64 和 arm64。

一、命名空间

命名空间：VDDataRead。包含 VDMocapSDK_DataRead_DataType.h 文件中的所有枚举类型及结构体，以及 VDMocapSDK_DataRead.h 文件中的所有成员函数。

二、VDMocapSDK_DataRead_DataType.h 文件说明

1. 枚举类型说明

1.1 _BodyNodes_

enum _BodyNodes_ ： uint8 [strong].

说明：表明全身 23 个节点及其排列序号。

枚举值：

0	BN_Hips	腰节点（Hips） —— 根节点
1	BN_RightUpperLeg	右大腿（Right Hip） —— 父节点：BN_Hips
2	BN_RightLowerLeg	右小腿（Right Knee） —— 父节点：BN_RightUpperLeg
3	BN_RightFoot	右脚掌（Right Ankle） —— 父节点：BN_RightLowerLeg
4	BN_RightToe	右脚趾（Right Toe） —— 父节点：BN_RightFoot
5	BN_LeftUpperLeg	左大腿（Left Hip） —— 父节点：BN_Hips
6	BN_LeftLowerLeg	左小腿（Left Knee） —— 父节点：BN_LeftUpperLeg
7	BN_LeftFoot	左脚掌（Left Ankle） —— 父节点： BN_LeftLowerLeg
8	BN_LeftToe	左脚趾（Left Toe） —— 父节点：BN_LeftFoot
9	BN_Spine	第一根脊椎（Chest） —— 父节点：BN_Hips

10	BN_Spine1	第二根脊椎（Chest1） —— 父节点：BN_Spine
11	BN_Spine2	第三根脊椎（Chest2） —— 父节点：BN_Spine1
12	BN_Spine3	第四根脊椎（Chest3） —— 父节点： BN_Spine2
13	BN_Neck	脖子（Neck） —— 父节点：BN_Spine3
14	BN_Head	头（Head） —— 父节点：BN_Neck
15	BN_RightShoulder	右肩（Right Collar） —— 父节点：BN_Spine3
16	BN_RightUpperArm	右大臂（Right Shoulder） —— 父节点：BN_RightShoulder
17	BN_RightLowerArm	右小臂（Right Elbow） —— 父节点：BN_RightUpperArm
18	BN_RightHand	右手掌(Right Wrist) —— 父节点: BN_RightLowerArm
19	BN_LeftShoulder	左肩（Left Collar） —— 父节点：BN_Spine3
20	BN_LeftUpperArm	左大臂(Left Shoulder) —— 父节点: BN_LeftShoulder
21	BN_LeftLowerArm	左小臂（Left Elbow） —— 父节点：BN_LeftUpperArm
22	BN_LeftHand	左手掌（Left Wrist） —— 父节点：BN_LeftLowerArm

1.2 _HandNodes_

enum _HandNodes_ ： uint8 [strong].

说明：表明单手手掌及手指共 20 个节点及其排列序号。

枚举值：

0	HN_Hand	手掌（手腕关节） —— 手指根节点

1	HN_ThumbFinger	大拇指根节（手掌中） —— 父节点：HN_Hand
2	HN_ThumbFinger1	大拇指第一节 —— 父节点：HN_ThumbFinger
3	HN_ThumbFinger2	大拇指第二节(最后节) —— 父节点:HN_ThumbFinger1
4	HN_IndexFinger	食指根节（手掌中） —— 父节点：HN_Hand
5	HN_IndexFinger1	食指第一节 —— 父节点：HN_IndexFinger
6	HN_IndexFinger2	食指第二节 —— 父节点：HN_IndexFinger1
7	HN_IndexFinger3	食指第三节（最后节） —— 父节点： HN_IndexFinger2
8	HN_MiddleFinger	中指根节（手掌中） —— 父节点：HN_Hand
9	HN_MiddleFinger1	中指第一节 —— 父节点：HN_MiddleFinger
10	HN_MiddleFinger2	中指第二节 —— 父节点：HN_MiddleFinger1
11	HN_MiddleFinger3	中指第三节（最后节） —— 父节点: HN_MiddleFinger2
12	HN_RingFinger	无名指根节（手掌中） —— 父节点： HN_Hand
13	HN_RingFinger1	无名指第一节 —— 父节点：HN_RingFinger
14	HN_RingFinger2	无名指第二节 —— 父节点：HN_RingFinger1
15	HN_RingFinger3	无名指第三节（最后节） —— 父节点：HN_RingFinger2
16	HN_PinkyFinger	小指根节（手掌中） —— 父节点：HN_Hand
17	HN_PinkyFinger1	小指第一节 —— 父节点：HN_PinkyFinger
18	HN_PinkyFinger2	小指第二节 —— 父节点：HN_PinkyFinger1
19	HN_PinkyFinger3	小指第三节（最后节） —— 父节点：HN_PinkyFinger2

1.3 _FaceBlendShapeARKit_

enum _FaceBlendShapeARKit_ ： uint8 [strong].

说明：52 个 ARKit BlendShape 表情参数

枚举值：

0	ARKIT_BrowDownLeft
1	ARKIT_BrowDownRight
2	ARKIT_BrowInnerUp
3	ARKIT_BrowOuterUpLeft
4	ARKIT_BrowOuterUpRight
5	ARKIT_CheekPuff
6	ARKIT_CheekSquintLeft
7	ARKIT_CheekSquintRight
8	ARKIT_EyeBlinkLeft
9	ARKIT_EyeBlinkRight
10	ARKIT_EyeLookDownLeft
11	ARKIT_EyeLookDownRight
12	ARKIT_EyeLookInLeft
13	ARKIT_EyeLookInRight
14	ARKIT_EyeLookOutLeft
15	ARKIT_EyeLookOutRight
16	ARKIT_EyeLookUpLeft

17	ARKIT_EyeLookUpRight
18	ARKIT_EyeSquintLeft
19	ARKIT_EyeSquintRight
20	ARKIT_EyeWideLeft
21	ARKIT_EyeWideRight
22	ARKIT_JawForward
23	ARKIT_JawLeft
24	ARKIT_JawOpen
25	ARKIT_JawRight
26	ARKIT_MouthClose
27	ARKIT_MouthDimpleLeft
28	ARKIT_MouthDimpleRight
29	ARKIT_MouthFrownLeft
30	ARKIT_MouthFrownRight
31	ARKIT_MouthFunnel
32	ARKIT_MouthLeft
33	ARKIT_MouthLowerDownLeft
34	ARKIT_MouthLowerDownRight
35	ARKIT_MouthPressLeft
36	ARKIT_MouthPressRight
37	ARKIT_MouthPucker
38	ARKIT_MouthRight

39	ARKIT_MouthRollLower
40	ARKIT_MouthRollUpper
41	ARKIT_MouthShrugLower
42	ARKIT_MouthShrugUpper
43	ARKIT_MouthSmileLeft
44	ARKIT_MouthSmileRight
45	ARKIT_MouthStretchLeft
46	ARKIT_MouthStretchRight
47	ARKIT_MouthUpperUpLeft
48	ARKIT_MouthUpperUpRight
49	ARKIT_NoseSneerLeft
50	ARKIT_NoseSneerRight
51	ARKIT_TongueOut

1.4 _FaceBlendShapeAudio_

enum _FaceBlendShapeAudio_ ： uint8 [strong].

说明：语音字母参数。

枚举值：

	AUDIO_a
	AUDIO_b
	AUDIO_c

	AUDIO_d
	AUDIO_e
	AUDIO_f
	AUDIO_g
	AUDIO_h
	AUDIO_i
	AUDIO_j
	AUDIO_k
	AUDIO_l
	AUDIO_m
	AUDIO_n
	AUDIO_o
	AUDIO_p
	AUDIO_q
	AUDIO_r
	AUDIO_s
	AUDIO_t
	AUDIO_u
	AUDIO_v
	AUDIO_w
	AUDIO_x
	AUDIO_y

	AUDIO_z

1.5 _SensorState_

enum _SensorState_ ： uint8 [strong].

说明：表示传感器工作状态。

枚举值：

0	SS_NONE	无
1	SS_Well	传感器工作正常
2	SS_NoData	传感器无数据，应检查是否接入传感器
3	SS_UnReady	传感器在启动中，应在空旷地方保持静止
4	SS_BadMag	传感器检测到异常磁场，应远离

1.6 _WorldSpace_

enum _WorldSpace_ ： uint8 [strong].

说明：表示模型所处世界坐标系。

枚举值：

0	WS_Geo	地理坐标系
1	WS_Unity	Unity 世界坐标系
2	WS_UE4	UE4 世界坐标系

1.7 _GESTURE_

enum _Gesture_ ： uint8 [strong].

说明：表示手势识别出的姿势。

枚举值：

0	GESTURE_NONE	未知手势(默认值)
1	GESTURE_1	食指伸直，其它手指握拢（指向）
2	GESTURE_2	剪刀手
3	GESTURE_3	OK
4	GESTURE_4	四
5	GESTURE_5	掌（布）
6	GESTURE_6	六
7	GESTURE_7	七
8	GESTURE_8	九
9	GESTURE_9	手枪
10	GESTURE_10	暂无
11	GESTURE_11	比心
12	GESTURE_12	大拇指、食指、小指伸直，其它手指握拢（爱你）
13	GESTURE_13	摇滚
14	GESTURE_14	赞
15	GESTURE_15	抓（拿）
16	GESTURE_16	握拳（石头）

17	GESTURE_17	手枪
18	GESTURE_18	暂无
19	GESTURE_19	踩
20	GESTURE_20	竖中指
21	GESTURE_21	竖尾指
22	GESTURE_22	三

2. 结构体说明

2.1 _MocapData_

struct _MocapData_.

说明：该结构体用于存储获取的动捕广播数据。

结构体包含项：

bool isUpdate	True：当前最新数据；false：无效数据
int frameIndex	帧序号：0-255
int frequency	设备数据传输频率，单位 HZ
int nsResult	其它数据
SensorState sensorState_body[NODES_BODY]	身体各节点处传感器工作状态，节点序号按照枚举_BodyNodes_排列
float position_body[NODES_BODY][3]	身体各节点坐标（xyz），单位 m，节点序号按照枚举_BodyNodes_排列
float quaternion_body[NODES_BODY][4]	身体各节点四元数（wxyz），节点序号按照

	枚举_BodyNodes_排列
float gyr_body[NODES_BODY][3]	身体各节点角速度，节点序号按照枚举_BodyNodes_排列
float acc_body[NODES_BODY][3]	身体各节点去重力加速度，节点序号按照枚举_BodyNodes_排列
float velocity_body[NODES_BODY][3]	身体各节点线速度，节点序号按照枚举_BodyNodes_排列
SensorState sensorState_rHand[NODES_HAND]	右手各节点处传感器工作状态，节点序号按照枚举_HandNodes_排列
float position_rHand[NODES_HAND][3]	右手各节点坐标（xyz），单位 m，节点序号按照枚举_HandNodes_排列
float quaternion_rHand[NODES_HAND][4]	右手各节点四元数（wxyz），节点序号按照枚举_HandNodes_排列
float gyr_rHand[NODES_HAND][3]	右手各节点角速度，节点序号按照枚举_HandNodes_排列
float acc_rHand[NODES_HAND][3]	右手各节点去重力加速度，节点序号按照枚举_HandNodes_排列
float velocity_rHand[NODES_HAND][3]	右手各节点线速度，节点序号按照枚举_HandNodes_排列
SensorState sensorState_lHand[NODES_HAND]	左手各节点处传感器工作状态，节点序号按照枚举_HandNodes_排列
float position_lHand[NODES_HAND][3]	左手各节点坐标（xyz），单位 m，节点序号

	按照枚举_HandNodes_排列
float quaternion_lHand[NODES_HAND][4]	左手各节点四元数（wxyz），节点序号按照枚举_HandNodes_排列
float gyr_lHand[NODES_HAND][3]	左手各节点角速度，节点序号按照枚举_HandNodes_排列
float acc_lHand[NODES_HAND][3]	左手各节点去重力加速度，节点序号按照枚举_HandNodes_排列
float velocity_lHand[NODES_HAND][3]	左手各节点线速度，节点序号按照枚举_HandNodes_排列
bool isUseFaceBlendShapesARKit	是否使用 ARKit BlendShapes
bool isUseFaceBlendShapesAudio	是否使用 Audio BlendShapes
float faceBlendShapesARKit[NODES_FACEBS_ARKIT]	ARKit BlendShapes 信息
float faceBlendShapesAudio[NODES_FACEBS_AUDIO]	Audio BlendShapes 信息
float localQuat_RightEyeball[4]	右眼四元数（wxyz）
float localQuat_LeftEyeball[4]	左眼四元数（wxyz）
Gesture gestureResultL	左手手势结果
Gesture gestureResultR	右手手势结果

2.2 _Version_

struct _Version_.

说明：该结构体用于存储版本信息。

结构体包含项：

uint8_t Project_Name[26]	工程名称
uint8_t Author_Organization[128]	开发者机构
uint8_t Author_Domain[26]	开发者网站
uint8_t Author_Maintainer[26]	维护联系邮箱
uint8_t Version[26]	版本号
uint8_t Version_Major	版本号 – 大改
uint8_t Version_Minor	版本号 – 小改
uint8_t Version_Patch	版本号 – 补丁

三、VDMocapSDK_DataRead.h 文件说明

该文件包含广播数据获取函数接口。以下是该文件中包含的成员函数。

1. GetVersionInfo()

成员函数	VDMOCAPSDKDATAREAD_API void GetVersionInfo(_Version_ *version);
说明	获取版本信息。
注意	_Version_ 在 "VDMocapSDK_DataRead_DataType.h" 文件中定义。

2. UdpOpen()

成员函数	VDMOCAPSDKDATAREAD_API bool UdpOpen(int index, unsigned short localPort)
说明	打开本地端口。
输入	index — 对象编号，默认使用 0，若需要读取多个广播对象，应使用不同 ID 接收。
输入	localPort — 本地端口
返回	True：成功，false：失败。

3. UdpClose()

成员函数	VDMOCAPSDKDATAREAD_API void UdpClose(int index);

说明	关闭由"Open"函数打开的本地端口。
输入	Index — 对象编号，默认使用 0，若需要读取多个广播对象，应使用不同 ID 接收。

4. UdpIsOpen()

成员函数	VDMOCAPSDKDATAREAD_API bool UdpIsOpen(int index);
说明	判断 udp 本地端口是否打开
输入	index — 对象编号，默认使用 0，若需要读取多个广播对象，应使用不同 ID 接收。
返回	True：开启状态，false：关闭状态。

5. UdpRemove()

成员函数	VDMOCAPSDKDATAREAD_API bool UdpRemove(int index, const char* dst_ip, unsigned short dst_port);
说明	移除已连接的远程端。
输入	index — 对象编号，默认使用 0，若需要读取多个广播对象，应使用不同 ID 接收。
输入	dst_ip — 远程端 ip 地址
输入	dst_port — 远程端端口
返回	True：成功，false：失败。

6. UdpSetPositionInInitialTpose()

成员函数	VDMOCAPSDKDATAREAD_API bool UdpSetPositionInInitialTpose(int index, const char* dst_ip, unsigned short dst_port, _WorldSpace_ worldSpace, float initialPosition_body[NODES_BODY][3], float initialPosition_rHand[NODES_HAND][3], float initialPosition_lHand[NODES_HAND][3]);
说明	设置 Tpose 骨架，若不设置，则默认使用读取到的骨架。
输入	index — 对象编号，默认使用 0，若需要读取多个广播对象，应使用不同 ID 接收。
输入	dst_ip — 远程端 ip 地址
输入	dst_port — 远程端端口
输入	worldSpace — 模型所在的世界坐标系
输入	initialPosition_body — 模型初始 Tpose 下各节点坐标，且按照枚举"_BodyNodes_"来排序
输入	initialPosition_rHand — 模型初始 Tpose 下右手各节点坐标，且按照枚举"_HandNodes_"来排序
输入	initialPosition_lHand — 模型初始 Tpose 下左手各节点坐标，且按照枚举"_HandNodes_"来排序

返回	True：成功，false：失败。
注意	_WorldSpace_ 在 "VDMocapSDK_DataRead_DataType.h" 文件中定义。 如果不需要设置骨架，不运行该接口即可，程序会使用动捕引擎的默认骨架。 UdpOpen()前后均可进行设置。 所述 T_pose：模型站立，双腿平行，且双脚脚掌都指向正前方，左右手分别向左右平举，掌心向下。

7. UdpSendRequestConnect()

成员函数	VDMOCAPSDKDATAREAD_API bool UdpSendRequestConnect(int index, const char* dst_ip, unsigned short dst_port);
说明	发送连接请求。
输入	index — 对象编号，默认使用 0，若需要读取多个广播对象，应使用不同 ID 接收。
输入	dst_ip — 远程端 ip 地址
输入	dst_port — 远程端端口
返回	True：连接成功，false：连接失败。

8. UdpRecvMocapData()

成员函数	VDMOCAPSDKDATAREAD_API bool UdpRecvMocapData(int index,

	const char* dst_ip, unsigned short dst_port, _MocapData_* mocapData);
说明	获取远程端发送过来的动捕数据。 获取广播数据后在新的模型上驱动（位移不一定相同） 若未设置骨架，则得到的是地理坐标系下的数据。 建议：广播的模型和接收的模型不同的时候使用
输入	index — 对象编号，默认使用 0，若需要读取多个广播对象，应使用不同 ID 接收。
输入	dst_ip — 远程端 ip 地址
输入	dst_port — 远程端端口
输出	mocapData — 动捕数据的结构体。
返回	True：成功，false：失败。
注意	结构体"_MocapData_"在头文件"VDMocapSDK_DataRead_DataType.h"中定义。

9. UdpGetRecvInitialTposePosition()

成员函数	VDMOCAPSDKDATAREAD_API bool UdpGetRecvInitialTposePosition(int index, const char *dst_ip, unsigned short dst_port,

	WorldSpace worldSpace, float initialPosition_body[NODES_BODY][3], float initialPosition_rHand[NODES_HAND][3], float initialPosition_lHand[NODES_HAND][3]);
说明	获取由 udp 接收到的模型初始 Tpose 各节点坐标。
输入	Index — 对象编号，默认使用 0，若需要读取多个广播对象，应使用不同 ID 接收。
输入	dst_ip — 远程端 ip 地址
输入	dst_port — 远程端端口
输入	worldSpace — 模型所在的世界坐标系
输出	initialPosition_body — 模型初始 Tpose 下各节点坐标，且按照枚举"_BodyNodes_"来排序
输出	initialPosition_rHand — 模型初始 Tpose 下右手各节点坐标，且按照枚举"_HandNodes_"来排序
输出	initialPosition_lHand — 模型初始 Tpose 下左手各节点坐标，且按照枚举"_HandNodes_"来排序
返回	True：成功，false：失败。
注意	枚举"_BodyNodes_"及"_HandNodes_"在头文件"VDMocapSDK_DataRead_DataType.h"中定义。 所述 T_pose：模型站立，双腿平行，且双脚脚掌都指向正前方，左右手分别向左右平举，掌心向下。

（完）