

Yanshee 仿真平台使用手册 V1.0

版本日志

日期	版本	修改内容	修订人
2022 年 4 月 10 日	V1.0	首次发布	李少春

目录

版本日志	1
目录	2
第 1 章 Yanshee 与 Webots 简介	4
1.1 Yanshee	4
1.2 Webots	4
1.3 Yanshee 的物理属性与功能	4
1.3.1 整体外观	4
1.3.2 硬件位置及接口	5
1.3.3 硬件参数规格表	6
1.3.4 Yanshee 能力架构	8
第 2 章 仿真平台使用	9
2.1 软硬件环境配置与安装	9
2.1.1 主机软硬件环境	9
2.1.2 Ros 安装与使用	9
2.1.3 Opencv 安装与使用	9
2.1.4 Webots 版本与安装说明	9
2.2 仿真平台工程文件包使用说明	10
2.2.1 仿真平台工程文件包下载地址	10
2.2.2 工程文件内容说明	10
2.2.3 工程文件包使用步骤	10
第 3 章 用户 ROS 接口说明	11
3.1 运动控制接口	11
3.1.1 设置舵机角度话题	11
3.1.2 读取舵机角度服务	12
3.1.3 步态控制服务	12
3.1.4 motion 动作文件控制	13
3.2 传感器接口	13
3.2.1 左红外传感器控制	14
3.2.2 右红外传感器控制	14
3.2.3 IMU 运动传感器控制	15
3.3 灯效设置服务	15
3.4 视觉识别接口	16
3.4.1 摄像头视频流	16
3.4.2 拍照接口服务	16
3.4.3 人脸识别	17
3.4.4 人脸属性（年龄、性别、眼镜、口罩、表情）	17
3.4.5 颜色识别	18
3.4.6 手势识别	19
3.4.7 物体识别	20
3.4.8 AprilTag 二维码识别	21
3.4.9 QR 二维码识别	22

第 4 章 用户 YanAPI 接口说明	24
4.1 运动控制接口	24
4.1.1 设置舵机角度	24
4.1.2 读取舵机角度	25
4.1.3 步态控制接口	26
4.1.4 motion 动作文件控制	26
4.2 传感器接口	27
4.2.1 获取传感器名称列表	27
4.2.2 读取红外距离传感器	28
4.2.3 读取 IMU 运动传感器	28
4.3 灯效控制接口	29
4.4 视觉识别接口	29
4.4.1 拍照接口	29
4.4.2 录入人脸	30
4.4.3 人脸识别	30
4.4.4 人脸属性（年龄、性别、眼镜、口罩、表情）	31
4.4.5 颜色识别	32
4.4.6 手势识别	32
4.4.7 物体识别	33
4.4.8 AprilTag 二维码识别	34
4.4.9 QR 二维码识别	35
第 5 章 示例与 Demo	36

第 1 章 Yanshee 与 Webots 简介

1.1 Yanshee

Yanshee 是优必选面向教育领域倾情打造的一款开源人形机器人教学平台。

它采用 Raspberry Pi + STM32 开放式硬件平台架构，17 个自由度的高度拟人设计，内置 800 万像素摄像头、陀螺仪及多种通信模块，配套多种开源传感器包，提供专业开源学习软件，支持 Blockly、Python、Java、C/C++ 等多种编程语言学习及多种 AI 应用的学习和开发。

它是一个帮助学生学习 AI 算法、编程、机器人运动学、仿真、ROS、物联网、人机互动等相关知识的极好平台。

提供专属课程、教材、以及专业的教研团队的授课培训，支持多种自定义赛事设计，是一套完整的多维度的开源机器人与 AI 教育解决方案！



1.2 Webots

Webots 是基于 OpenGL 与 ODE (Open Dynamics Engine) 的开源机器人仿真模拟器，支持 ROS, C/C++, python 等接口，具备丰富的机器人模型、场景与传感器，通过设计合理的 ROS 接口，可以与硬件平台实现无缝切换。Webots 不仅能够精确的描述机器人运动的物理规律并渲染逼真丰富的环境信息，而且能够准确的描述现实中的多种传感器，与机器人及环境的动态变化过程。

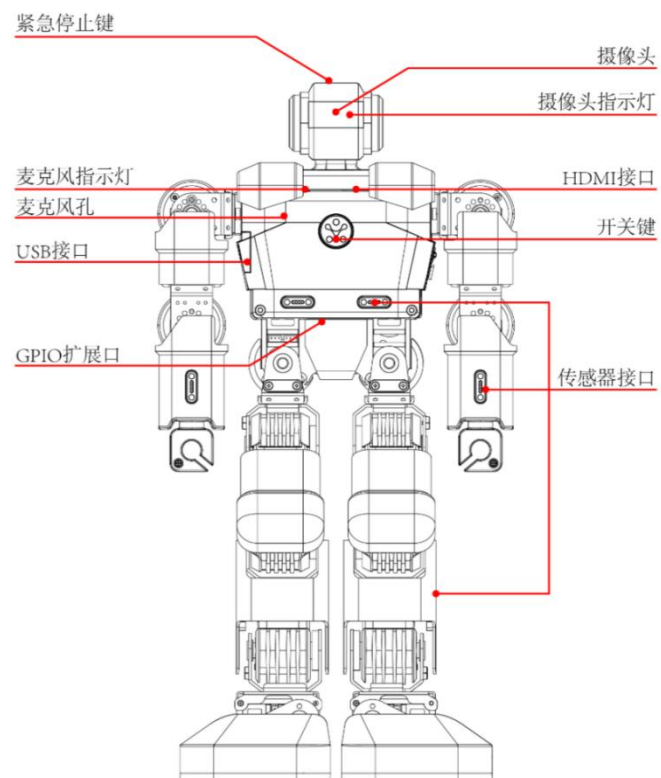
1.3 Yanshee 的物理属性与功能

1.3.1 整体外观

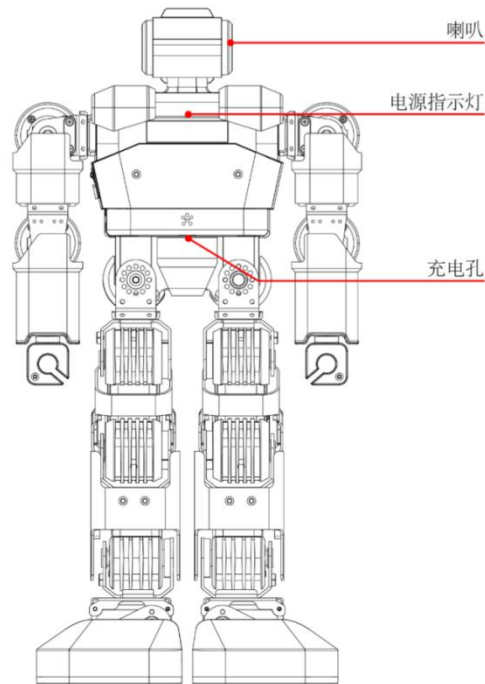
Yanshee 做为一款开源人形机器人，其整体外观如下图所示，其中蓝色区域代表它身上的 17 个自由度的舵机所在位置。



1.3.2 硬件位置及接口



正面



背面

1.3.3 硬件参数规格表

外观	
产品造型	人形外观
产品颜色	银色
产品尺寸	370*192*106 (mm)
产品重量	≈2.09kg
材质	铝合金结构、PC+ABS 外壳
伺服舵机	17 个自由度 高速带离合舵机
电气性能	
工作电压	DC 9.6V
功率	4.5W-38.4W
工作温度	0℃~40℃
电源适配器	输入:100V-240V~50/60Hz 1A 输出:9.6V, 4A
主芯片及存储器	
处理器	STM32F103VDT6+ Broadcom BCM2837 1.2GHz 64-bit quad-core ARMv8 Cortex-A53 (Raspbian Pi 3B)

内存	1GB
存储	16GB
操作系统	Raspbian
网络	
Wifi	支持 Wi-Fi2.4G 802.11b/g/n 快速连接
蓝牙	蓝牙 4.1
电池容量	3100mAh

视觉	
摄像头	800 万像素，定焦
灯光	眼：三色 LED 灯 *2 胸灯光：三色 LED 呼吸灯 *3 麦克风灯：绿色指示灯*1 充电：双色指示灯*1
音频	
麦克风	单麦克风
喇叭	立体声喇叭 *2
传感器	
内置传感器	九轴运动控制（Motion Tracking）传感器 *1 主板温度检测传感器 *1
扩展接口	POGO 4PIN *6
调试接口	
HDMI	1
GPIO	40（6 个已占用）
USB	2
其它	
按键	胸口电源键头顶紧急制动按键
控制方式	手机 APP 语音控制

1.3.4 Yanshee 能力架构

1 运动控制

17 自由度伺服舵机，内置 MCU，包含伺服控制、传感反馈及直流驱动系统。

支持 Webots、Gazebo 运动仿真、正逆运动控制、步态算法行走。内置俯卧撑、格斗、舞蹈表演等丰富动作库，同时支持自定义 PRP 回读编程动作并使用。

2 传感系统

支持四种 POGO 磁吸接口传感器：红外、温湿度、触摸、压力传感器。可随意组合，构成多样场景设计。内置九轴 IMU 陀螺仪传感器，实时反馈姿态信息，机器人摔倒后可自动爬起。通过跨 MCU 设计获得传感器精准数据，用于多场景拓展教学。

3 机器视觉

搭载 800 万像素高清摄像头，支持机器人拍照、视频录制、网络视频流传输、视频监控等。支持人脸跟踪、检测、年龄、性别、表情、眼镜、口罩识别和物体识别等 AI 模型。可通过颜色识别、形状识别、二维码识别、目标检测等视觉功能获得环境信息并完成机器人多种竞赛方案设计。

4 语音识别

集成语音闲聊，强大的云服务后台支持。内置多种动作触发词，增强人机交互体验感。开放定制离线命令词，用户可根据需求自定义关键词，完成相应场景设计。语音识别、语义理解、TTS、语言翻译等多种 AI 算法模型，提供对应项目实践课程。

5 通信系统

ROS 系统通信。支持通过 ROS 节点进行基本的机器人功能模块通信，实现 ROS 编程控制机器人内部功能，并支持跨平台多机 ROS 通信。支持包括 UDP、TCP 组播广播等常用通信方式。支持机器人集控通信。物联网案例包括天气站、Web 控制、智能管家等强大的物联网 IoT 学习入口。

6 二次开发接口

二次开发接口 YanAPI 覆盖运动控制、设备信息、智能传感器、机器视觉、语音识别等模块。支持多语言开发的 RestfulAPI 接口，包括：C、C++、C#、PHP、Java、Curl、Python 等常用主流开发语言。默认 Python 脚本格式，并通过 Blockly 引导编程。支持用户接入第三方 AI 云服务 API 接口（例如：百度、微软、腾讯、Face++ 等），完成多种人工智能模型集成。

第 2 章 仿真平台使用

2.1 软硬件环境配置与安装

2.1.1 主机软硬件环境

建议主机硬件配置

CPU Intel i7 或以上, 显卡 GTX 950 或以上

推荐操作系统:

Ubuntu 版本 18.04 64 位

2.1.2 Ros 安装与使用

推荐 ros 版本:

ros_melodic

安装详情参考官方说明:

<http://wiki.ros.org/ROS/Installation>

2.1.3 Openvino 安装与使用

推荐安装版本:

openvino_2020.1.023

安装详情参考官网说明:

https://docs.openvino.ai/latest/openvino_docs_install_guides_installing_openvino_linux.html#install-openvino

2.1.4 Webots 版本与安装说明

推荐使用版本安装包:

webots_2021a_amd64.deb

下载路径:

<https://github.com/cyberbotics/webots/releases/tag/R2021a>

安装 webots-ros 控制器:

`sudo apt-get install ros-melodic-webots-ros`

2.2 仿真平台工程文件包使用说明

2.2.1 仿真平台工程包下载地址

https://github.com/shaoyiwork/yanshee_simulation_platform

2.2.2 工程文件内容说明

文件目录	内容说明
UBT-YansheeSimulation-Ros-Linux-v1.0.3-xxx.deb	用来安装 Yanshee 虚拟仿真机器人 ROS 服务的压缩包文件
deps	安装 yanshee 仿真环境的 ros 服务所需要的依赖包
user_demo/ros_demo	用户控制机器人所有功能的 ROS 接口使用 demo
user_demo/YanAPI_demo	用户控制机器人所有功能的 YanAPI 接口使用 demo
yanshee_simulation_project/worlds	webots 软件环境所需的 wbt 文件
yanshee_simulation_project/model	Yanshee 机器人的 wbo 模型文件
yanshee_simulation_project/controllers	Yanshee 的控制器执行文件
yanshee_simulation_project/scripts	用于将 hts 转成 motion 文件的脚本
yanshee_simulation_project/motion	存放 motion 动作文件的位置

2.2.3 工程文件包使用步骤

步骤 1、拷贝或克隆所有文件到 Ubuntu18.04 环境下。例如：/home/用户目录下。

步骤 2、使用命令安装 yanshee-ros 服务 deb 文件包

```
sudo dpkg -i UBT-YansheeSimulation-Ros-Linux-v1.0.3-xxx.deb
```

步骤 3、安装依赖包。命令行：cd deps 然后 ./install.sh 等待安装完成。

步骤 4、添加 yanshee-ros 服务到环境变量中，编辑 ~/.bashrc 文件最后一行添加

```
source /opt/yanshee/setup.bash
```

保存退出。然后使用命令 source .bashrc 使其生效。

步骤 5、另起窗口，输入 roscore，启动 ros 主程序。

步骤 6、另起窗口，输入如下命令，启动 yanshee-ros 服务。

```
roslaunch yanshee_launch yanshee.launch
```

步骤 7、另起窗口，命令行 webots 之后，打开 yanshee_simulation_project/worlds 目录下相应的 wbt 场景文件，点击运行按钮，即可开始仿真。

第 3 章 用户 ROS 接口说明

仿真平台 API 主要包括两大部分：用户 ROS 接口和用户 YanAPI 接口。其中 ROS 接口支持 Python、C++ 和 ros 命令三种方式调用，而 YanAPI 接口仅支持 Python3 接口调用。两种接口为不同仿真用户提供最大可能的拓展性和易用性。本章主要介绍用户 ROS 接口。

用户 ROS 接口主要包括运动控制、传感器、灯效、视觉四大部分。

3.1 运动控制接口

3.1.1 设置舵机角度话题

名称：/hal_servo_write

类型：ubt_msgs/servo_write_list

功能：用户通过发布该话题来达到控制机器人的一组舵机角度转动的目的。

传参说明：

ubt_msgs/servo_write[] data

符号	说明
data.name	关节名称
data.angle	设置角度
data.runtime	运行时间 ms

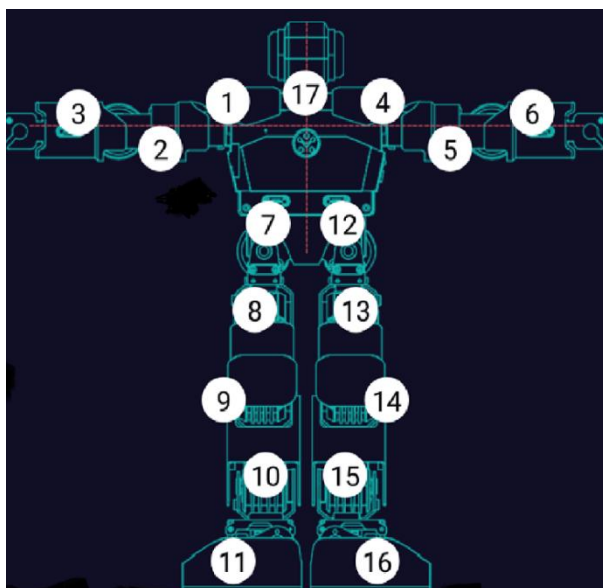
参考示例：

ros_demo/servo_test.py

关节名称与舵机序号对照表：

舵机序号	关节名称	舵机序号	关节名称
1	RightShoulderRoll	10	RightAnkleFB
2	RightShoulderFlex	11	RightAnkleUD
3	RightElbowFlex	12	LeftHipLR
4	LeftShoulderRoll	13	LeftHipFB
5	LeftShoulderFlex	14	LeftKneeFlex
6	LeftElbowFlex	15	LeftAnkleFB
7	RightHipLR	16	LeftAnkleUD
8	RightHipFB	17	NeckLR
9	RightKneeFlex		

舵机序号与实物对照图：



3.1.2 读取舵机角度服务

名称: /hal_servo_read

类型: ubt_msgs/servo_read_list

接口功能: 用户通过向该服务发送请求来读取当前一组舵机角度值。

传参说明:

string[] name : 舵机名称列表。

返回值说明:

ubt_msgs/servo_read[] data : 返回舵机角度信息列表。

其中每个成员包括

符号	说明
data.name	关节名称
data.angle	当前角度

参考示例:

ros_demo/servo_test.py

3.1.3 步态控制服务

名称: /set_gait_play

类型: ubt_msgs/gait_operation

接口功能: 用户通过向该服务发送请求来执行机器人的步态行走。

传参说明:

ubt_msgs/gait_para input

符号	说明
input.period	1-5 步频参数

input.speed_h	-5-0 向后走 0-5 向前走
input.speed_v	横向行走速度
input.steps	行走步数
input.wave	是否摇摆手臂
input.timestamp	Unix 时间戳

参考示例：

ros_demo/gait_test.py

3.1.4 motion 动作文件控制

1 查询 motion 动作文件列表服务

名称：/get_motion_list

类型：ubt_msgs/motion_list

接口功能：用户通过向该服务发送请求来查询机器人内置 motion 动作。

参考示例：

ros_demo/motion_test.py

2 开始执行 motion 动作服务

名称：/set_motion_play

类型：ubt_msgs/motion_play

接口功能：用户通过向该服务发送请求来执行机器人内置 motion 动作。

传参说明：

string name : 需要执行的动作名称。

int32 repeat : 需要执行的重复次数。

参考示例：

ros_demo/motion_test.py

3 停止执行 motion 动作服务

名称：/set_motion_stop

类型：ubt_msgs/motion_stop

接口功能：用户通过向该服务发送请求来停止正在做的 motion 动作。

传参说明：

string name : 需要停止的动作名称，为空时默认停止所有动作。

参考示例：

ros_demo/motion_test.py

3.2 传感器接口

传感器接口分为话题和服务两种，话题用来订阅，服务用来查询。并都有各自的使能服务触发接口。

3.2.1 左红外传感器控制

1 左红外距离传感器查询服务

名称: /distance_left_get_value

类型: ubt_msgs/get_float

接口功能: 用户通过向该服务发送请求来读取当前左红外传感器的值。

参考示例:

ros_demo/sensor_test.py

2 左红外距离传感器话题使能服务

名称: /distance_left_enable

类型: ubt_msgs/set_int

接口功能: 用户通过向该服务发送请求来使能左红外传感器话题发布。

3 左红外距离传感器话题

名称: /distance_left_value

类型: sensor_msgs/Range

接口功能: 用户通过订阅该话题来获取左红外传感器实时数据值。

3.2.2 右红外传感器控制

1 右红外距离传感器查询服务

名称: /distance_right_get_value

类型: ubt_msgs/get_float

接口功能: 用户通过向该服务发送请求来读取当前右红外传感器的值。

参考示例:

ros_demo/sensor_test.py

2 右红外距离传感器话题使能服务

名称: /distance_right_enable

类型: ubt_msgs/set_int

接口功能: 用户通过向该服务发送请求来使能右红外传感器话题发布。

3 右红外距离传感器话题

名称: /distance_right_value

类型: sensor_msgs/Range

接口功能: 用户通过订阅该话题来获取右红外传感器实时数据值。

3.2.3 IMU 运动传感器控制

1 IMU 运动传感器查询服务

名称: /get_imu_data

类型: ubt_msgs/imu_read

接口功能: 用户通过向该服务发送请求来读取当前 IMU 运动传感器的值。

参考示例:

ros_demo/sensor_test.py

2 IMU 运动传感器话题使能服务

名称: /report_imu_enable

类型: ubt_msgs/enable_set

接口功能: 用户通过向该服务发送请求来使能 IMU 运动传感器话题发布。

3 IMU 运动传感器话题

名称: /report_imu_data

类型: ubt_msgs/imu_report

接口功能: 用户通过订阅该话题来获取 IMU 运动传感器实时数据值。

例如 data = imu_report()

返回值说明表:

符号	说明
data.imu.linear_acceleration.x	绕 X 轴方向线加速度
data.imu.linear_acceleration.y	绕 Y 轴方向线加速度
data.imu.linear_acceleration.z	绕 Z 轴方向线加速度
data.imu.angular_velocity.x	绕 X 轴方向角速度 (gyro-x)
data.imu.angular_velocity.y	绕 Y 轴方向角速度 (gyro-y)
data.imu.angular_velocity.z	绕 Z 轴方向角速度 (gyro-z)
data.compass.magnetic_field.x	磁力计 x
data.compass.magnetic_field.y	磁力计 y
data.compass.magnetic_field.z	磁力计 z
data.euler.euler_x	欧拉角 x
data.euler.euler_y	欧拉角 y
data.euler.euler_z	欧拉角 z

3.3 灯效设置服务

名称: /set_led

类型: ubt_msgs/led_write

接口功能: 设置灯效包括眼灯 (三色三模式) 和胸前灯 (七色四模式)。

传参说明:

ubt_msgs/led data

符号	说明
data.name	灯名称。可填：button 胸前灯，camera 眼睛灯
data.mode	灯模式。当 name 为 button 时：on、off、blink、breath 当 name 为 camera 时：on、off、blink
data.value	灯颜色。当 name 为 button 时：white、red、green、blue、yellow、purple、cyan，当 name 为 camera 时：red、green、blue

参考示例：

ros_demo/led_test.py

3.4 视觉识别接口

3.4.1 摄像头视频流

1 摄像头视频流话题使能服务

名称：/camera_enable

类型：ubt_msgs/set_int

接口功能：用户通过向该服务发送请求来打开或关闭摄像头视频流话题。

参数说明：

Int32 value：等于 1 时，使能摄像头，等于 0 时关闭摄像头。

命令示例：

rosservice call /camera_enable "value: 1"

2 摄像头视频流话题

名称：/camera_image

类型：sensor_msgs/Image

接口功能：用户通过订阅该话题来获取摄像头视频流源数据帧。

3.4.2 拍照接口服务

名称：/camera_take_photo

类型：ubt_msgs/save_image

接口功能：用户通过向该服务发送请求来拍一张照片，只支持 jpg 和 png 格式。

参数说明：

符号	说明
filename	拍照名称全路径，例如 “/tmp/my_photo.jpg”
quality	拍照质量 1-100。仅支持 jpg

参考示例：

ros_demo/vision_take_photo.py

3.4.3 人脸识别

1 人脸识别任务操作服务

名称: /face_recognition_task

类型: ubt_msgs/vision_task_set

接口功能: 开启或停止人脸识别任务。

传参说明:

string data: 可填“start”或“stop”表示开始或停止人脸识别任务。

参考示例:

ros_demo/vision_face_rec.py

2 获取人脸识别结果服务

名称: /face_rec_result

类型: ubt_msgs/vision_result

接口功能: 获取人脸识别结果及其坐标值。

返回值说明:

符号	说明
result	识别到的人脸名称
local_info	人脸坐标值 (x,y,w,h 分别为左上角坐标值和识别框宽高值)
status	任务执行状态, idle 和 run 两种, 当为 idle 时可以读取结果
rc	等于 0 代表成功

参考示例:

ros_demo/vision_face_rec.py

3.4.4 人脸属性 (年龄、性别、眼镜、口罩、表情)

1 人脸属性任务操作服务

名称: /face_attribute_task

类型: ubt_msgs/vision_task_set

接口功能: 开启或停止人脸属性任务。

传参说明:

string data: 可填“start”或“stop”表示开始或停止人脸属性任务。

参考示例:

ros_demo/vision_face_attr.py

2 获取人脸属性结果服务

名称: /face_attr_result

类型: ubt_msgs/vision_face_attr_result

接口功能: 获取人脸属性结果值。

返回值说明:

符号	说明
result	人脸属性识别结果列表。 <code>face_attribute[]</code> 类型，具体成员见： <code>face_attribute</code> 类型成员说明表
status	任务执行状态， <code>idle</code> 和 <code>run</code> 两种。当为 <code>idle</code> 时可以读取结果。

`face_attribute` 类型成员说明表：

符号	说明
age	年龄 0-100
gender	性别（male female）
age_group	年龄段（baby、children、juvenile、youth、middle_age、old_age）
smile	微笑程度 0-100
glass	是否带眼镜（grayglass、normalglass、noglass）
mask	是否戴口罩（masked、unmasked、not masked well）
expression	表情（happy、surprise、normal）
local_info.x	人脸框左上角 x 坐标值
local_info.y	人脸框左上角 y 坐标值
local_info.w	人脸框宽度值
local_info.h	人脸框高度值

参考示例：

ros_demo/vision_face_attr.py

3.4.5 颜色识别

1 颜色识别任务操作服务

名称：/color_recognition_task

类型：ubt_msgs/vision_task_set

接口功能：开启或停止颜色识别任务。

vision_task_set.srv 参数说明：

string data: 可填“start”或“stop”表示开始或停止颜色识别任务。

参考示例：

ros_demo/vision_color_rec.py

2 获取颜色识别结果服务

名称：/color_rec_result

类型：ubt_msgs/vision_result

接口功能：获取颜色识别结果及其坐标值。

返回值说明：

符号	说明
result	识别到的颜色名称，详见：颜色返回值表
status	任务执行状态， <code>idle</code> 和 <code>run</code> 两种，当为 <code>idle</code> 时可以读取结果
local_info.x	颜色框左上角 x 坐标值

local_info.y	颜色框左上角 y 坐标值
local_info.w	颜色框宽度值
local_info.h	颜色框高度值

颜色返回值表：

返回值	说明	返回值	说明
none	无结果	magenta	洋红
pink	粉	orange	橙
red	红	violet	紫
green	绿	brown	棕
blue	蓝	black	黑
yellow	黄	white	白
cyan	青	gray	灰

参考示例：

ros_demo/vision_color_rec.py

3.4.6 手势识别

1 手势识别任务操作服务

名称：/gesture_recognition_task

类型：ubt_msgs/vision_task_set

接口功能：开启或停止手势识别任务。

vision_task_set.srv 参数说明：

string data: 可填“start”或“stop”表示开始或停止手势识别任务。

参考示例：

ros_demo/vision_gesture_rec.py

2 获取手势识别结果服务

名称：/gesture_rec_result

类型：ubt_msgs/vision_result_list

接口功能：获取手势识别结果及其坐标值。

返回值说明：

符号	说明
result	识别到的手势信息列表。vision_info[] 类型
status	任务执行状态，idle 和 run 两种，当为 idle 时可以读取结果

vision_info 类型成员表：

符号	说明
name	识别到的手势名称。具体见手势名称表。
local_info.x	手势框左上角 x 坐标值
local_info.y	手势框左上角 y 坐标值

local_info.w	手势框宽度值
local_info.h	手势框高度值

手势名称返回值表：

返回值	说明	返回值	说明
none	无结果	palm	手掌
awesome	帅气	pray	祈祷
bad	拇指朝下	ROCK	摇滚（拇、食、无名指）
call	打电话	salute	作揖
good	拇指朝上	scissor	剪刀
heartA	比心 A	stone	石头
heartB	比心 B	swear	发誓
heartC	比心 C	take_picture	拍照
OK	OK	thanks	谢谢
one	单指		

参考示例：

ros_demo/vision_gesture_rec.py

3.4.7 物体识别

1 物体识别任务操作服务

名称：/object_recognition_task

类型：ubt_msgs/vision_task_set

接口功能：开启或停止物体识别任务。

vision_task_set.srv 参数说明：

string data: 可填“start”或“stop”表示开始或停止物体识别任务。

参考示例：

ros_demo/vision_object_rec.py

2 获取物体识别结果服务

名称：/object_rec_result

类型：ubt_msgs/vision_result

接口功能：获取物体识别结果及其坐标值。

返回值说明：

符号	说明
result	识别到的物体名称，详见：物体名称表
status	任务执行状态，idle 和 run 两种，当为 idle 时可以读取结果
local_info.x	识别框左上角 x 坐标值
local_info.y	识别框左上角 y 坐标值
local_info.w	识别框宽度值
local_info.h	识别框高度值

物体名称返回值表：

返回值	说明	返回值	说明
none	无结果	azalea	映山红
hand	手掌	hibiscus	木槿
qr_code	二维码	gypsophila_paniculata	满天星
wukong_robot	悟空机器人	rose	玫瑰
mobile_phone	手机	lily	百合花
stuffed_animal	毛绒玩具	hyacinthus_orientalis	风信子
rubiks_cube	魔方	frangipani	鸡蛋花
carambola	杨桃	butterfly_orchid	蝴蝶兰属
peach	桃子	cosmos	大波斯菊
pear	梨	zinnia	百日菊
cherry	樱桃	daisy	雏菊
pitaya	火龙果	rohdea_japonica	万年青
avocado	牛油果	pachira_macrocarpa	发财树
kiwi	猕猴桃	chlorophytum_comosum	吊兰
mango	芒果	clivia	君子兰
apple	苹果	dracaena_sanderiana	富贵竹
strawberry	草莓	asparagus_setaceus	文竹
grapes	葡萄	narcissus_tazetta	水仙
banana	香蕉	epipremnum_aureum	绿萝
orange	橙子	sansevieria_trifasciata	虎皮兰
forget_me_not	勿忘草	zamioculcas	金钱树
sunflower	向日葵花	kalanchoe_blossfeldiana	长寿花
carnation	康乃馨	yanshee_robot	Yanshee 机器人

参考示例：

ros_demo/vision_object_rec.py

3.4.8 AprilTag 二维码识别

1 AprilTag 二维码识别任务操作服务

名称：/april_tag_detector_task

类型：ubt_msgs/ctrl_april_tag_continuous_detector

接口功能：开启或停止 AprilTag 二维码识别任务。

参数说明：

符号	说明
enable	使能识别任务。bool 类型
tags	需要识别的二维码信息列表。StandaloneTag[] 类型
enableStream	使能网络视频流。bool 类型

StandaloneTag 类型成员表:

符号	说明
id	TAG36H11 标准 id (取值: 0 - 586), int 类型
size	apirltag 边长 (单位: 米 m), float 类型

参考示例:

ros_demo/vision_apriltag.py

2 获取 AprilTag 二维码识别结果服务

名称: /april_tag_result

类型: ubt_msgs/check_april_tag_status

接口功能: 获取 AprilTag 二维码识别结果 (包括了二维码 id 值、三维坐标值和姿态四元素值)。

参数说明:

符号	说明
april_tag_status	二维码识别结果。april_tag_detection_array 类型
enable	使能成功与否, 当为 true 时可以读取数据。bool 类型

april_tag_detection_array 成员表:

例如: (data = april_tag_status.detections[0])

符号	说明
data .id[0]	识别到二维码的 id 值
data .pose.pose.pose.position.x	三维空间 x 坐标值
data .pose.pose.pose.position.y	三维空间 y 坐标值
data .pose.pose.pose.position.z	三维空间 z 坐标值
data .pose.pose.pose.orientation.x	位姿四元素 x 值
data .pose.pose.pose.orientation.y	位姿四元素 y 值
data .pose.pose.pose.orientation.z	位姿四元素 z 值
data .pose.pose.pose.orientation.w	位姿四元素 w 值

参考示例:

ros_demo/vision_apriltag.py

3.4.9 QR 二维码识别

1 QR 二维码识别任务操作服务

名称: /qr_detector_task

类型: ubt_msgs/ctrl_qr_detector

接口功能: 开启或停止 QR 二维码识别任务。

参数说明:

符号	说明
enable	使能识别任务。bool 类型
enableStream	使能网络视频流。bool 类型

参考示例：

ros_demo/vision_qr_detector.py

2 获取 QR 二维码识别结果服务

名称：/qr_detector_result

类型：ubt_msgs/check_qr_status

接口功能：获取 QR 二维码识别结果。

参数说明：

符号	说明
contents	返回的二维码识别结果内容列表。string[]类型
enable	使能成功与否，当为 true 时可以读取数据。bool 类型

参考示例：

ros_demo/vision_qr_detector.py

第 4 章 用户 YanAPI 接口说明

为了便于移植与使用，本系列接口保持与实际 Yanshee 机器人真机一致的参数与方法。目前主要支持运动控制、传感器、灯效、视觉识别等四大部分。基本上该系列接口可以认为是真机 YanAPI 的一个裁剪版本。

4.1 运动控制接口

4.1.1 设置舵机角度

```
YanAPI.set_servos_angles(angles: Dict[str, int], runtime: int = 20)
```

功能：设置舵机角度值（一次可以设置一个或者多个舵机角度值）

参数说明：

angles (dict) - {servoName:angle}, servoName (str) -机器人舵机名称, angle (int) -舵机角度值

runtime (int) - 运行时间，单位：毫秒 ms，取值范围：20~4000

舵机名称及角度范围表：

servoName	angle 可设置范围	servoName	angle 可设置范围
RightShoulderRoll	0-180	RightAnkleFB	0-180
RightShoulderFlex	0-180	RightAnkleUD	65-180
RightElbowFlex	0-180	LeftHipLR	60-180
LeftShoulderRoll	0-180	LeftHipFB	0-170
LeftShoulderFlex	0-180	LeftKneeFlex	0-180
LeftElbowFlex	0-180	LeftAnkleFB	0-180
RightHipLR	0-120	LeftAnkleUD	0-115
RightHipFB	10-180	NeckLR	0-180
RightKneeFlex	0-180		

返回类型 dict

返回说明：

```
{
  code:integer 返回码：0 表示正常
  data:
    {
      RightShoulderRoll:bool  True-设置成功， False-设置失败
      RightShoulderFlex:bool  True-设置成功， False-设置失败
      RightElbowFlex:bool     True-设置成功， False-设置失败
      LeftShoulderRoll:bool   True-设置成功， False-设置失败
```



```

        LeftShoulderFlex:bool    True-设置成功, False-设置失败
        LeftElbowFlex:bool       True-设置成功, False-设置失败
        RightHipLR:bool          True-设置成功, False-设置失败
        RightHipFB:bool          True-设置成功, False-设置失败
        RightKneeFlex:bool       True-设置成功, False-设置失败
        RightAnkleFB:bool        True-设置成功, False-设置失败
        RightAnkleUD:bool        True-设置成功, False-设置失败
        LeftHipLR:bool           True-设置成功, False-设置失败
        LeftHipFB: bool          True-设置成功, False-设置失败
        LeftKneeFlex:bool        True-设置成功, False-设置失败
        LeftAnkleFB:bool         True-设置成功, False-设置失败
        LeftAnkleUD:bool         True-设置成功, False-设置失败
        NeckLR:bool              True-设置成功, False-设置失败
    }
    msg:string 提示信息
}

```

参考示例:

YanAPI_demo/servos_test.py

4.1.2 读取舵机角度

YanAPI.get_servos_angles(names: List[str])

功能: 查询舵机角度值 (一次可以查询一个或者多个舵机角度值)

参数说明:

names list[string] - 机器人舵机名称列表

返回类型 dict

返回说明:

```

{
    code:integer 返回码: 0 表示正常
    data:
    {
        RightShoulderRoll:90    1 号舵机
        RightShoulderFlex: 90   2 号舵机
        RightElbowFlex: 90      3 号舵机
        LeftShoulderRoll: 90     4 号舵机
        LeftShoulderFlex:90      5 号舵机
        LeftElbowFlex: 90        6 号舵机
        RightHipLR: 90           7 号舵机
        RightHipFB: 60           8 号舵机
        RightKneeFlex:76         9 号舵机
        RightAnkleFB:110         10 号舵机
    }
}

```

```

        RightAnkleUD: 90      11 号舵机
        LeftHipLR: 90        12 号舵机
        LeftHipFB:120        13 号舵机
        LeftKneeFlex:104     14 号舵机
        LeftAnkleFB: 70      15 号舵机
        LeftAnkleUD:90       16 号舵机
        NeckLR: 90           17 号舵机
    }
    msg: string 提示信息
}

```

参考示例：

YanAPI_demo/servos_test.py

4.1.3 步态控制接口

YanAPI.sync_do_motion_gait(speed_v: int = 0, speed_h: int = 0, steps: int = 0, period: int = 1, wave: bool = False)

功能：机器人步态动作控制，执行完成后返回

参数说明：

period (int) - 取值【0~5】，period = 0 表示停止步态

speed_v (int) - 前后垂直行走速度，取值【-5~5】

speed_h (int) - 左右水平行走速度，取值【-5~5】

steps (int) - 总步数值，大于零的正整数。默认值 steps=0 代表 10 亿这样一个极大值。

wave (bool) - 表示是否开启手臂摆动，取值 True/False，默认值为 False 表示不开启手臂摆动

返回说明：类型 bool，True-执行成功，False-执行失败

参考示例：

YanAPI_demo/gait_test.py

4.1.4 motion 动作文件控制

1 获取动作文件列表

YanAPI.get_motion_list_value()

返回值说明：

返回动作文件列表，类型 List： 例如：[A, B, C, D, E, F, G]

参考示例

YanAPI_demo/motion_test.py

2 同步执行动作

YanAPI.sync_play_motion(name: str = “reset”, repeat: int = 1)

功能：开始执行动作，执行完成后返回

参数说明：

name (str) - 动作文件名

repeat (int) - 重复次数: 1-100

返回值说明: 类型 bool, True-执行成功, False-执行失败

参考示例

YanAPI_demo/motion_test.py

3 异步执行动作

YanAPI.start_play_motion(name: str = "reset", repeat: int = 1)

功能: 开始执行动作, 立即返回

参数说明:

name (str) - 动作文件名

repeat (int) - 重复次数: 1-100

返回值说明:

Dict:

```
{
    code:integer (int32)返回码, 0 表示正常
    data:
        {
            total_time:integer (int32)运行完成需要的时间 (单位 ms)
        }
    msg:string 提示信息
}
```

参考示例

YanAPI_demo/motion_test.py

4 停止执行动作

YanAPI.stop_play_motion(name: str = "")

功能: 停止动作执行

参数说明:

name (str) - 动作文件名, 当 name=""表示停止所有动作

参考示例

YanAPI_demo/motion_test.py

4.2 传感器接口

4.2.1 获取传感器名称列表

YanAPI.get_sensors_list_value()

获取当前机器人在使用的传感器名称列表。

返回值说明:

传感器名称列表, 类型 list。

例如: ['imu', 'distance_left', 'distance_right']

4.2.2 读取红外距离传感器

YanAPI.get_sensors_infrared_value()

功能：直接返回单个红外传感器的值。

返回值说明：int, 0-1500。

YanAPI.get_sensors_infrared()

功能：返回所有红外传感器的距离值。

返回值说明：

```
{
    code:integer 返回码, 0 表示正常
    data:
        {
            infrared:
                [
                    {
                        name:str 传感器名称
                        value:int 距离值, 单位: 毫米 mm
                    }
                ]
        }
    msg:string 提示信息
}
```

参考示例

YanAPI_demo/ sensor_test.py

4.2.3 读取 IMU 运动传感器

YanAPI.get_sensors_gyro()

功能：获取九轴陀螺仪运动传感器值

返回值说明：

```
{
    code:integer 返回码: 0 表示正常
    data:
        {
            gyro:[
                {
                    id:integer 传感器 ID 值, 取值: 1~127
                    gyro-x:(float) 陀螺仪 x
                    gyro-y:(float) 陀螺仪 y
                    gyro-z:(float) 陀螺仪 z
                    accel-x:(float) 加速度计 x
                    accel-y:(float) 加速度计 y
                    accel-z:(float) 加速度计 z
                }
            ]
        }
}
```

```

        compass-x:(float)      磁力计 x
        compass-y:(float)      磁力计 y
        compass-z:(float)      磁力计 z
        euler-x:(float)        欧拉角 x
        euler-y:(float)        欧拉角 y
        euler-z:(float)        欧拉角 z
    }
}
msg:string 提示信息
}

```

参考示例

YanAPI_demo/ sensor_test.py

4.3 灯效控制接口

YanAPI.set_robot_led(type: str, color: str, mode: str)

功能：设置灯效包括眼灯（三色三模式）和胸前灯（七色四模式）

参数说明：

符号	说明
type	灯类型。可填：button 胸前灯，camera 眼睛灯
color	灯颜色。当 type 为 button 时：white、red、green、blue、yellow、purple、cyan，当 type 为 camera 时：red、green、blue
mode	灯模式。当 type 为 button 时：on、off、blink、breath 当 type 为 camera 时：on、off、blink

参考示例

YanAPI_demo/ led_test.py

4.4 视觉识别接口

4.4.1 拍照接口

YanAPI.take_vision_photo(resolution: str = '640x480')

功能：拍一张照片，默认存储路径为/tmp/photo

参数说明：

resolution (str) - 照片分辨率，默认拍照分辨率为“640x480”，最大拍照分辨率为“1920x1080”

返回说明：

```

{
    code:integer 返回码：0 表示正常
    data:
    {

```

```

        name:string 照片文件名称
    }
    msg:string 提示信息
}

```

参考示例

YanAPI_demo/ vision_take_photo.py

4.4.2 录入人脸

YanAPI.do_face_entry(name)

功能：录入人脸数据到人脸数据库中，并打 tag。

参数说明：

name(str) - 为人脸数据命名的 tag 名称。

返回说明：

类型 bool ， True-执行成功， False-执行失败。

参考示例

YanAPI_demo/ face_input.py

4.4.3 人脸识别

1 综合接口：

YanAPI.sync_do_face_recognition("recognition")

功能：返回识别到的人脸名称及坐标值。

返回值说明：

```

{
    code: int 返回码： 0 表示正常
    type:string : recognition。
    data:
        {
            recognition: {
                name: string 识别到的人脸名称
            }
            location: {
                x: int 识别框左上角 x 坐标值
                y: int 识别框左上角 y 坐标值
                w: int 识别框宽度值
                h: int 识别框高度值
            }
        }
    status: string 状态
    msg: string 提示信息
}

```

参考示例

YanAPI_demo/ face_recognition.py

2 简单接口：

YanAPI.sync_do_face_recognition_value("recognition")

功能：直接返回识别到的人脸名称。

返回值说明：类型 str，识别到的人脸名称。

参考示例

YanAPI_demo/ face_recognition.py

4.4.4 人脸属性（年龄、性别、眼镜、口罩、表情）

1 综合接口：

YanAPI.sync_do_face_recognition(type)

功能：识别人脸相关属性的值及人脸坐标值。

传参说明：

type--需要识别的类型，可填：quantity | age_group | gender | age | expression | mask | glass

返回值说明：

```
{
    code: integer 返回码：0 表示正常
    type: string 消息类型，一次只返回一种类型的数据。
    data:
        {
            analysis: {
                age: int 年龄
                group: string 年龄段
                gender: string 性别
                expression: string 表情：happy/surprise/normal
                mask: string 口罩识别结果：masked/unmasked/not masked well
                glass: string 眼镜识别结果：grayglass/normalglass/noglass
            }
            quantity: int 人脸数量
            location: {
                x: int 识别框左上角 x 坐标值
                y: int 识别框左上角 y 坐标值
                w: int 识别框宽度值
                h: int 识别框高度值
            }
        }
    status: string 状态
    msg: string 提示信息
}
```

参考示例

YanAPI_demo/face_attribute.py

2 简单接口：

YanAPI.sync_do_face_recognition_value(type)

功能：识别人脸相关属性的值。

传参说明：

type--需要识别的类型，可填：quantity | age_group | gender | age | expression | mask | glass

返回值说明：

直接返回对应属性的识别值。

参考示例

YanAPI_demo/face_attribute.py

4.4.5 颜色识别

YanAPI.sync_do_color_recognition()

功能：识别返回当前看到的最大色块的颜色。

返回值说明：

```
{
    code: integer 返回码：0 表示正常
    data:
        {
            color: {
                name: string 识别到的颜色名称
            }
            location: {
                x: int 识别框左上角 x 坐标值
                y: int 识别框左上角 y 坐标值
                w: int 识别框宽度值
                h: int 识别框高度值
            }
        }
    type: color_detect
    status: string 状态
    msg: string 提示信息
}
```

参考示例

YanAPI_demo/color_recognition.py

4.4.6 手势识别

YanAPI.sync_do_gesture_recognition()

功能：返回当前识别到的手势及坐标值。

返回值说明：

```
{
  code: integer 返回码：0 表示正常
  data:
    {
      gesture: string 识别到的手势名称（详见：手势名称返回值表）
      location: {
        x: int 识别框左上角 x 坐标值
        y: int 识别框左上角 y 坐标值
        w: int 识别框宽度值
        h: int 识别框高度值
      }
    }
  type: gesture
  status: string 状态
  msg: string 提示信息
}
```

参考示例

YanAPI_demo/gesture_recognition.py

4.4.7 物体识别

YanAPI.sync_do_object_recognition()

功能：返回当前识别到的物体及坐标值。

返回值说明：

```
{
  code: integer 返回码：0 表示正常
  data:
    {
      recognition: {
        name: string 识别到的物体名称（详见：物体名称返回值表）
      }
      location: {
        x: int 识别框左上角 x 坐标值
        y: int 识别框左上角 y 坐标值
        w: int 识别框宽度值
        h: int 识别框高度值
      }
    }
  type: recognition
  status: string 状态
  msg: string 提示信息
}
```

```
}
```

参考示例

YanAPI_demo/object_recognition.py

4.4.8 AprilTag 二维码识别

1 开始二维码识别：

YanAPI.start_aprilTag_recognition(tags: List, enableStream: bool = False)

功能：开始 aprilTag 识别

参数说明：

tags (List[dict]) - 需要识别的 apriltagID 和边长信息。

```
[
    {
        id: int      TAG36H11 标准 id (取值： 0 - 586)
        size:float   apirltag 边长（单位：米 m）
    }
]
```

enableStream (bool) - 是否需要打开视频流，默认值为 False 表示不开启视频流

返回值说明：

```
{
    code: integer 返回码： 0 表示正常
    streamUrl: string 视频流 URL 信息
    msg: string 提示信息
}
```

参考示例：

YanAPI_demo/aprilTag_test.py

2 获取二维码结果：

YanAPI.get_aprilTag_recognition_status()

功能：查询 aprilTag 识别结果

返回说明：

```
{
    code: integer 返回码： 0 表示正常
    data:
        {
            AprilTagStatus 识别到的 apriltag 数组
            [
                {
                    id: integer AprilTag id
                    # 相对摄像头的位姿（用四元数及三维坐标表示）
                    orientation-x:integer,
                    orientation-y:integer,
```

```

        orientation-z:integer,
        orientation-w:integer,
        position-x:integer,
        position-y:integer,
        position-z:integer
    }
}
]
}
status: string 执行状态: idle-非执行状态 run-正在运行
msg: string 提示信息
}

```

参考示例:

YanAPI_demo/aprilTag_test.py

3 停止二维码识别:

YanAPI.stop_aprilTag_recognition()

功能: 停止 aprilTag 识别

返回值说明:

```

{
    code: integer 返回码: 0 表示正常
    streamUrl: string 视频流 URL 信息
    msg: string 提示信息
}

```

参考示例:

YanAPI_demo/stop_aprilTag.py

4.4.9 QR 二维码识别

YanAPI.sync_do_QR_code_recognition(timeout: int)

功能: 开始二维码识别, 识别完成后返回

参数说明:

timeout (int) - 最大等待时间 (单位: 秒 s), ≤ 0 表示直到识别成功后停止

返回值说明:

```

{
    code: integer 返回码: 0 表示正常
    content: [string] 识别到的内容列表
    status: string 状态
    msg: string 提示信息
}

```

参考示例:

YanAPI_demo/qr_detector.py

第 5 章 示例与 Demo

请参考 user_demo 文件夹下相应的 demo 内容。该系列 demo 全部采用 python3 编程语言编写。用户可在命令行直接通过 `python3 xxx.py` 来运行和验证相关接口功能。并根据提供的基础 Demo 来设计自己需要实现的比赛场景或其它综合使用场景 Demo。

至此平台使用手册介绍完毕，大家赶快试一试吧，预祝玩的愉快！