一、系统

1.设置整机运动

Function: robot_ctrl.set_mode(mode_enum)

Parameters:

- mode_enum(enum)
  - rm_define.robot_mode_gimbal_follow
  - rm_define.robot_mode_chassis_follow
  - rm_define.robot_mode_free

2.控制计时器开始、暂停或结束计时

Function: tools.timer_ctrl(ehavior_enum)

Parameters:

- behavior_enum(enum):
  - rm_define.timer_start
  - rm_define.timer_stop
  - rm_define.timer_reset

3.放大相机倍镜，局部图像更清晰

Function: media_ctrl.zoom_value_update(value)

Parameters:

- value (int): [1, 4]

4.信息类（变量型数据）获取计时器从开始到当前时刻的用时，返回秒数

Function: tools.timer_current()

Return value:

- time_stamp(float)

5.信息类（变量型数据）获取程序运行用时，返回秒数

Function: tools.run_time_of_program()

Return value:

- time (float)

6.信息类（变量型数据）获取当前的时间信息，如年/月/日/时/分/秒等。

Function: tools.get_localtime(time_enum)

Parameters:

- time_enum (enum):
  - ■rm_define.localtime_year
  - ■rm_define.localtime_month
  - ■rm_define.localtime_day
  - ■rm_define.localtime_hour
  - ■rm_define.localtime_minute
  - ■rm_define.localtime_second

Return value

- time (int)

7.信息类（变量型数据）机器人启动时刻至今的时间间隔，返回累计的秒数，1）机器人的启动时刻是指上电时刻。2）如果机器人在断电后重启，会重新累计时间戳。

Function: tools.get_unixtime()

Return value:

- time (float)

二、灯效

1.设置指定位置 LED 灯的闪烁频率，2Hz 即每秒闪烁 2 次

Function: led_ctrl.set_flash(armor_enum, frequency)

Parameters:

- armor_enum(enum):
  - rm_define.armor_all
  - rm_define.armor_bottom_front
  - rm_define.armor_bottom_back
  - rm_define.armor_bottom_left
  - rm_define.armor_bottom_right
  - rm_define.armor_top_left
  - rm_define.armor_top_right
- frequency(int): [1, 10]

2.控制底盘指定位置 LED 灯的颜色和灯效：

- 常亮，LED 灯保持点亮状态
- 熄灭，LED 灯关闭
- 呼吸，LED 灯明暗变化（由暗变亮再变暗）
- 闪烁，LED 灯以一定频率闪烁

Function: led_ctrl.set_bottom_led(armor_enum, r, g, b,

led_effect_enum)

Parameters:

- armor_enum(enum):
  - rm_define.armor_bottom_all
  - rm_define.armor_bottom_front
  - rm_define.armor_bottom_back
  - rm_define.armor_bottom_left
  - rm_define.armor_bottom_right
- r(int): [0, 255]
- g(int): [0, 255]
- b(int): [0, 255]
- led_effect_enum(enum):
  - rm_define.effect_always_on
  - rm_define.effect_always_off
  - rm_define.effect_breath
  - rm_define.effect_flash

3.设置云台指定位置 LED 灯的颜色和灯效：

- 常亮，LED 灯保持点亮状态
- 熄灭，LED 灯关闭
- 呼吸，LED 灯明暗变化（由暗变亮再变暗）
- 闪烁，LED 灯以一定频率闪烁
- 跑马灯，呈圆形排布的 8 颗 LED 灯顺时针滚动点亮

Function: led_ctrl.set_top_led(armor_enum, r, g, b, led_effect_enum)

Parameters:

- armor_enum(enum):
  - rm_define.armor_top_all
  - rm_define.armor_top_left
  - rm_define.armor_top_right
- r(int): [0, 255]
- g(int): [0, 255]
- b(int): [0, 255]
- led_effect_enum(enum):
  - rm_define.effect_always_on
  - rm_define.effect_always_off
  - rm_define.effect_breath
  - rm_define.effect_flash
  - rm_define.effect_marquee

4.设置云台指定序号 LED 灯的亮灭，序号 1～8 分别对应云台两侧可独立控制的 8 颗 LED 灯

Function: led_ctrl.set_signle_led(armor_enum, led_index, led_effect_enum)

Parameters:

- armor_enum(enum):

        - rm_define.armor_top_all
        - rm_define.armor_top_left
        - rm_define.armor_top_right
    - index(int/list): [1, 8]
    - led_effect_enum(enum):
        - rm_define.effect_always_on
        - rm_define.effect_always_off

5.关闭指定位置的 LED 灯

Function: led_ctrl.turn_off(armor_enum)

Parameters:

- armor_enum(enum)
    - rm_define.armor_all
    - rm_define.armor_bottom_front
    - rm_define.armor_bottom_back
    - rm_define.armor_bottom_left
    - rm_define.armor_bottom_right
    - rm_define.armor_top_left
    - rm_define.armor_top_right

6.控制发射器弹道灯的亮灭

Function: led_ctrl.gun_led_on()

led_ctrl.gun_led_off()

三、底盘

1.设置 PWM 输出百分比，数值越大，在某一周期内高电平的持续时间越长。该 PWM 基础频率为 50Hz（灯的亮灭、舵机转动）

注意：

1）PWM 口位于底盘控制模块上，拿开底盘后侧的透明盖板即可看到。从上至下共 6 个 PWM 口。

2）PWM 又称脉冲宽度调制，控制的是某一周期内高电平的持续时间，现广泛应用于 LED 灯、舵机等的控制上。

3）上电后，PWM 接口默认输出 7.5%占空比的信号，每次程序运行结束后，也会恢复默认的输出信号。

4）对灯条来说，PWM 输出百分比范围为 0%~100%，0 意味着灯最暗，100 意味着灯最亮。

5）对舵机来说，PWM 输出百分比范围为 2.5% ~ 12.5%。因为大部分舵机的控制脉冲频率为 50 Hz，控制周期为 20 ms，可调节角度-90 °~ 90° 对应的高电平脉宽为 0.5 ms ~ 2.5 ms， 因此舵机占空比的控制范围便是 0.5/20~2.5/20，即 2.5% ~ 12.5%。

玩家们可以根据自己想要控制的旋转角度设置舵机 PWM 的输出百分比。

Function: chassis_ctrl.set_pwm_value(pwm_port_enum, output_percent)

Parameters:

- pwm_port_enum(enum)
  - rm_define.pwm_all
  - rm_define.pwm1
  - rm_define.pwm2
  - rm_define.pwm3
  - rm_define.pwm4
  - rm_define.pwm5
  - rm_define.pwm6
- output_percent(int): [0, 100]

2.开启或关闭底盘速度杆量叠加

注意：

1）如果不添加“开启底盘速度杆量叠加”模块，在运行程序时我们无法手控底盘；而添加此模块后，我们就可以对运行中的机器人进行移动控制，并且控制量会叠加。

2）杆量是指摇杆幅值的推动大小。杆量范围为 -1 ~ 1。

3）速度杆量叠加是将底盘在程序中的速度与摇杆速度相加。

在程序中设置底盘以 0.5m/s 向前平移，同时我们将杆量推满， 开启底盘速度杆量叠加后，机器人就会将两种控制数据“叠加”， 最终机器人将以（0.5+1*当前最大向前速度）m/s 的速率向前平移。

Function: chassis_ctrl.enable_stick_overlay()

chassis_ctrl.disable_stick_overlay()

3. 在“底盘跟随云台模式”下，当云台左右旋转时，底盘始终与云台保持指定夹角

Function: chassis_ctrl. set_follow_gimbal_offset(degree)

Parameters:

- degree(int): [-180, 180]°

4.设置底盘平移速率，默认平移速率是 0.5 米/秒。数值越大，移动越快。

Function: chassis_ctrl.set_trans_speed(speed)

Parameters:

- speed(float): [0, 3.5] m/s

5.设置底盘旋转速率，默认旋转速率是 30 度/秒。数值越大，旋转越快。

Function: chassis_ctrl.set_rotate_speed(speed)

Parameters:

- speed(int): [0, 600] °/s

6.独立控制四个麦轮的转速，符合麦轮转动方向和速度的有效组合才会生效。

Function: chassis_ctrl.set_wheel_speed(lf_speed, rf_speed, lr_speed, rr_speed)

Parameters:

- lf_speed(int): [-1000, 1000] rpm
- rf_speed(int): [-1000, 1000] rpm
- lr_speed(int): [-1000, 1000] rpm
- rr_speed(int): [-1000, 1000] rpm

7.控制底盘向指定方向平移

Function: chassis_ctrl.move(degree)

Parameters:

- degree (int): [-180, 180] °

8.控制底盘向指定方向平移指定时长

Function: chassis_ctrl.move_with_time(degree, time)

Parameters:

- degree(int): [-180, 180] °
- time(float): [0, 20] s

注：此方法比控制底盘向某方向平移并等待几秒、以及控制底盘向某方向平移多少米的精度都要高，且走的要平顺一些

9.控制底盘向指定方向平移指定距离

Function: chassis_ctrl.move_with_distance(degree, distance)

Parameters:

- degree(int): [-180, 180] °

- distance(float): [0, 5] m

10.控制底盘以指定的平移速率向指定方向平移

Function: chassis_ctrl.move_degree_with_speed(speed, degree)

Parameters:

- speed(float): [0, 3.5] m/s
- degree(int): [-180, 180] °

11.控制底盘向指定方向旋转

Function: chassis_ctrl.rotate(direction_enum)

Parameters:

- direction_enum(enum):
  - rm_define.clockwise
  - rm_define.anticlockwise

12. 控制底盘向指定方向旋转指定时长

Function: chassis_ctrl.rotate_with_time(direction_enum, time)

Parameters:

- direction_enum(enum):
  - rm_define.clockwise
  - rm_define.anticlockwise
- time(float): [0, 20] s

13.控制底盘向指定方向旋转指定角度

Function: chassis_ctrl.rotate_with_degree(direction_enum, degree)

Parameters:

- direction_enum(enum):
  - rm_define.clockwise #常数：6
  - rm_define.anticlockwise #常数：5
- degree(int): [0, 1800] °

注释：顺时针正 60 不等于逆时针负 60，参数只能是负数，不会将负值自动转换方向的正值

14.控制底盘向指定方向平移的同时做旋转运动

Function: chassis_ctrl.move_and_rotate(degree, direction)

Parameters:

- degree(int): [-180, 180] °
- direction_enum(enum):
  - rm_define.clockwise
  - rm_define.anticlockwise

15.控制底盘以指定速度在指定方向运动

Function: chassis_ctrl.move_with_speed(speed_x, speed_y, speed_rotation)

Parameters:

- speed_x(float): [0, 3.5] m/s
- speed_y(float): [0, 3.5] m/s
- speed_rotation(int): [-600, 600] °/s

16.停止底盘的所有运动

Function: chassis_ctrl.stop()

17.信息类（变量型数据）

以上电时刻底盘位置为基准，获取底盘当前在航向轴、俯仰轴或翻滚轴上的姿态角值

Function: chassis_ctrl.get_attitude(attitude_enum)

Parameters:

- attitude_enum(enum):
  - rm_define.chassis_yaw
  - rm_define.chassis_pitch
  - rm_define.chassis_roll

Return value:

- degree(float)

18. 信息类（变量型数据）

获取底盘当前位置的坐标和朝向数据，x、y 的方向以上电时枪管方向为基准（故如果上电时枪管方向和程序运行时的方向垂直的话，得出的结果就是

正好颠倒的，应该加以注意），但（0，0）点是以程序开始运行时的底盘中心点为基准的。s1 测得的坐标、朝向信息和实测基本相等，精度较高。

Function: chassis_ctrl.get_position_based_power_on(action_enum)

Parameters:

- action_enum(enum):
  - rm_define.chassis_forward #x 向，上电时枪管的方向
  - rm_define.chassis_translation #y 向，上电时右侧板方向
  - rm_define.chassis_rotate #朝向，以程序开始运行时的枪管方向为基准

Return value:

- position(float)

19. 事件类(中断处理程序)

在行驶过程中，当底盘撞击到人、桌腿等障碍物时，运行本模块内程序

Function: def chassis_impact_detection(msg)

Type: Event callback

20. 布尔型(真伪判别)

在行驶过程中，检测到底盘撞击到人、桌腿等障碍物时会返回“真”，否则返回“假”

Function:chassis_ctrl.is_impact()

Return value:

- impact_status(bool)

四、云台

1.开启或关闭云台速度杆量叠加

Function: gimbal_ctrl.enable_stick_overlay()

gimbal_ctrl.disable_stick_overlay()

2.在“云台跟随底盘模式”下，当底盘左右旋转时，云台始终与底盘保持指定夹角

Function: gimbal_ctrl.set_follow_chassis_offset(degree)

Parameters:

- degree(int): [-180, 180] °

3.设置云台旋转速率，默认速率是 30 度/秒。数值越大，旋转越快。

Function: gimbal_ctrl.set_rotate_speed(speed)

Parameters:

- speed(float): [0, 540] °/s

4.控制云台动作：

- 回中：云台回到航向轴和俯仰轴的初始位置
- 停止运动：云台停止运动，但仍处于受控状态
- 休眠：云台断电

- 唤醒：云台重新通电

休眠时云台疲软，电机无力矩输出，可以被随意推动；唤醒后云台恢复控制，自动回中

Function: gimbal_ctrl.recenter()

gimbal_ctrl.stop()

gimbal_ctrl.suspend()

gimbal_ctrl.resume()

5.控制云台向指定方向旋转(注：非阻塞型，可以继续后面的动作，相当于同步运行的)

此模块会控制云台向指定方向持续旋转，直到收到“控制云台停止运动”、“等待（x）秒”或其它控制云台运动的指令

Function: gimbal_ctrl.rotate(direction_enum)

Parameters:

- direction_enum(enum):
  - rm_define.gimbal_up
  - rm_define.gimbal_down
  - rm_define.gimbal_left
  - rm_define.gimbal_right

6.控制云台向指定方向旋转指定角度（注：阻塞型，故后面不用跟休眠时间以给其运行时间）

Function: gimbal_ctrl.rotate_with_degree(direction_enum, degree)

Parameters:

- direction_enum(enum):
  - rm_define.gimbal_up
  - rm_define.gimbal_down
  - rm_define.gimbal_left
  - rm_define.gimbal_right
- degree(int): [0, 55]°

注释：试验证明，向上旋转正 20 度与向下旋转负 20 度的效果一样，可以自动识别正负和方向。向左旋转正 60 度与向右旋转负 60 度的效果一样

7.控制云台绕航向轴旋转到指定位置

Function: gimbal_ctrl.yaw_ctrl(degree)

Parameters:

- degree(int): [-250, 250]°

8.控制云台绕俯仰轴旋转到指定位置

Function: gimbal_ctrl.pitch_ctrl(degree)

Parameters:

- degree(int): [-20, 35]°

9.控制云台旋转到指定角度位置

注意：

1）在“云台跟随底盘模式”下，“控制云台旋转到航向轴（x）度”部分不生效，“旋转到俯仰轴（x）度”部分不受影响。

2）“控制云台（向上/下/左/右）旋转（x）度”是相对位置，基于云台当前方位。

3）“控制云台绕航向轴旋转到（x）度”、“控制云台绕俯仰轴旋转到（x）度”、“控制云台旋转到航向轴（x）度 俯仰轴（x）度”是绝对位置，基于底盘当前方位。

Function: gimbal_ctrl.angle_ctrl(yaw_degree, pitch_degree)

Parameters:

- yaw_degree (int): [-250, 250]°
- pitch_degree (int): [-20, 35]°

10.控制云台以指定旋转速度同时绕航向轴、俯仰轴旋转

注意：

速度输入值的正负号代表着云台转动方向。

对航向轴而言：正值意味着向右转动，负值意味着向左转动。

对俯仰轴而言：正值意味着向上转动，负值意味着向下转动。

Function: gimbal_ctrl.rotate_with_speed(yaw_speed, pitch_speed)

Parameters:

- yaw_speed(float): [-360, 360]°/s

- pitch_speed(float): [-360, 360]°/s

11.信息类（变量型数据）

获取云台当前在航向轴或俯仰轴上的姿态角值，以当前底盘朝向为基准

Function: gimbal_ctrl.get_axis_angle(axis_enum)

Parameters:

- axis_enum (enum):
  - rm_define.gimbal_axis_yaw　　#航向角
  - rm_define.gimbal_axis_pitch　　#俯仰角

Return value:

- degree(int)

五、发射器

1.设置发弹数，即每次射出的水弹颗数

Function: gun_ctrl.set_fire_count(count)

Parameters:

- count(int): [1, 8]

2.控制发射器只发射一次水弹(执行类、阻塞型)

Function: gun_ctrl.fire_once()

3.控制发射器持续发射水弹(执行类、非阻塞型)

注意：

1）默认频率是每秒发射一次，每次发弹一颗。

2）连续发射水弹是非阻塞型模块，也就是它会持续发射，直到遇到“停止发射”指令或例程结束的情况。

Function: gun_ctrl.fire_continuous()

4.停止发射水弹

注意：

1）因为“单次发射水弹”模块是阻塞型的，所以“停止发射水弹”指令对它不起作用。

2）“停止发射水弹”模块只对“连续发射水弹”有限制。

Function: gun_ctrl.stop()

六、智能

1.开启或关闭 RoboMaster S1 对视觉标签、姿势、同类 S1 机器人、行人、线的视觉识别功能

注意：

1）RoboMaster S1 的智能识别功能默认处于关闭状态。所以如果未先开启识别功能，机器人对相应的可识别信息就不会有反应。

2）机器人默认能够识别的线颜色是蓝色。

Function: vision_ctrl.enable_detection(function_enum)

vision_ctrl.disable_detection(function_enum)

Parameters:

- function_enum(enum):
  - rm_define.vision_detection_marker
  - rm_define.vision_detection_pose
  - rm_define.vision_detection_car
  - rm_define.vision_detection_people
  - rm_define.vision_detection_line

2.开启或关闭掌声识别功能

Function: media_ctrl.enable_sound_recognition(function_enum)

media_ctrl.disable_sound_recognition(function_enum)

Parameters:

- function_enum(enum):
  - rm_define.sound_detection_applause

3.设置视觉标签的有效识别距离，超出限制则不会识别

Function: vision_ctrl.set_marker_detection_distance(distance)

Parameters:

- distance(float): [0.5, 3]

4.设置机器人能够识别的线颜色

Function: vision_ctrl.line_follow_color_set(color_enum)

Parameters:

- color_enum(enum):
  - rm_define.line_follow_color_blue
  - rm_define.line_follow_color_red
  - rm_define.line_follow_color_green

5.在巡线时减小曝光值，可以缓解快速转向造成的视野模糊，让识别更稳定。

Function:

media_ctrl.exposure_value_update(exposure_value_enum)

Parameters:

- exposure_value_enum(enum):
  - rm_define.exposure_value_large
  - rm_define.exposure_value_medium
  - rm_define.exposure_value_small

6.机器人识别并瞄准对应视觉标签的中心位置

注意：

1)请先“开启视觉标签识别”功能，否则不会识别。

2)使用此模块时，当视野中有红心出现就会瞄准，5 秒内没有识别到红心的话会超时退出，运行后面的程序。

Function: vision_ctrl.detect_marker_and_aim(marker_enum)

Parameters:

- rm_define.marker_trans_red_heart #红心
- rm_define.marker_trans_target #靶子
- rm_define.marker_trans_dice #骰子
- rm_define.marker_number_[zero,..., nine]

注：rm_define.marker_number_zero 为常数 10，one-nine 对应常数 11-19

- rm_define.marker_letter_[A,..., Z]

7.事件类(中断处理程序)

当识别到物体、视觉标签、姿势等对应信息时运行本模块内的程序

注意：

1）事件触发模块优先级高，也就是说无论主线程运行到哪一步，只要满足事件触发条件，就会立刻跳出主线程，开始运行事件类模块内的程序。

2）需识别行人时，请控制云台适度向上旋转，让行人站在机器人前 1 米处，不要蹲着，确保行人完整出现在机器人视野中。

3）V 字、倒 V 字的姿势指令需要用手臂完成，而非手指。

Function: def vision_recognized_people(msg)

def vision_recognized_car(msg) #S1

def vision_recognized_pose_all(msg) #任一姿势

def vision_recognized_pose_victory(msg)# V 型

def vision_recognized_pose_give_in(msg)#倒 V 型

def vision_recognized_pose_capture(msg)#拍照手势

def vision_recognized_marker_trans_all(msg)#任一方向

def vision_recognized_marker_trans_left(msg)

def vision_recognized_marker_trans_right(msg)

def vision_recognized_marker_trans_stop(msg)

def vision_recognized_marker_trans_forward(msg)

def vision_recognized_marker_trans_red_heart(msg)

def vision_recognized_marker_trans_target(msg)#靶子

def vision_recognized_marker_trans_dice(msg)#骰子

def vision_recognized_marker_number_all(msg)#任一数字

def vision_recognized_marker_number_[one, …, nine](msg)

def vision_recognized_marker_letter_all(msg)#任一字母

def vision_recognized_marker_letter_[A, …, Z](msg)

Type: Event callback

8.事件类(中断处理程序)

当识别到对应的拍手指令时将运行本模块内的程序

Function: def sound_recognized_applause_twice(msg)

def sound_recognized_applause_thrice(msg)

Type: Event callback

9.布尔型(多和条件类模块连用)

识别到物体、视觉标签、姿势、拍手指令等对应信息时返回“真”，否则将返回“假”

Function: vision_ctrl.check_condition(condition_enum)

Parameters:

- condition_enum(enum):
  - rm_define.cond_recognized_people
  - rm_define.cond_recognized_car
  - rm_define.cond_recognized_marker_trans_all
  - rm_define.cond_recognized_marker_trans_left
  - rm_define.cond_recognized_marker_trans_right
  - rm_define.cond_recognized_marker_trans_forward
  - rm_define.cond_recognized_marker_trans_stop
  - rm_define.cond_recognized_marker_trans_red_heart
  - rm_define.cond_recognized_marker_trans_target
  - rm_define.cond_recognized_marker_trans_dice
  - rm_define.cond_recognized_marker_number_all
  - rm_define.cond_recognized_marker_number_[zero,..., nine]
  - rm_define.cond_recognized_marker_letter_all
  - rm_define.cond_recognized_marker_letter_[A,..., Z]

- rm_define.cond_recognized_pose_all
- rm_define.cond_recognized_pose_victory
- rm_define.cond_recognized_give_in
- rm_define.cond_recognized_capture
- rm_define.cond_sound_recognized_applause_twice #两次拍手
- rm_define.cond_sound_recognized_applause_thrice #三次

10.待机器人识别到物体、视觉标签、姿势等对应信息时将继续执行，否则将持续等待(执行类、阻塞型)

Function: vision_ctrl.cond_wait(condition_enum)

Parameters:

- condition_enum(enum):
  - rm_define.cond_recognized_people
  - rm_define.cond_recognized_car
  - rm_define.cond_recognized_marker_trans_all
  - rm_define.cond_recognized_marker_trans_left
  - rm_define.cond_recognized_marker_trans_right
  - rm_define.cond_recognized_marker_trans_forward
  - rm_define.cond_recognized_marker_trans_stop
  - rm_define.cond_recognized_marker_trans_red_heart

- rm_define.cond_recognized_marker_trans_target
- rm_define.cond_recognized_marker_trans_dice
- rm_define.cond_recognized_marker_number_all
- rm_define.cond_recognized_marker_number_[zero,..., nine]
- rm_define.cond_recognized_marker_letter_all
- rm_define.cond_recognized_marker_letter_[A,..., Z]
- rm_define.cond_recognized_pose_all
- rm_define.cond_recognized_pose_victory
- rm_define.cond_recognized_give_in
- rm_define.cond_recognized_capture
- rm_define.cond_sound_recognized_applause_twice
- rm_define.cond_sound_recognized_applause_thrice

11.信息类（列表型数据）注：是 python 的列表，下标从零开始

获取识别到的视觉标签信息，参数为 N , ID , X , Y , W, H

注意：

1）视觉标签信息的格式为：

第 1 项为识别到的视觉标签数量 N，后续项以 5 个数据为一组，分别是第一个视觉标签的 ID，标签中心点在机器人视野中位置的横坐标 X、纵坐标 Y、宽度 W 和高度 H；第二个视觉标签的 ID、标签中心点在机器

人视野中位置的横坐标、纵坐标、宽度和高度；第三个...

2）返回的 ID 值释义：

ID 值为 1 代表识别到的是：停止

ID 值为 2 代表识别到的是：骰子

ID 值为 3 代表识别到的是：靶

ID 值为 4 代表识别到的是：左箭头

ID 值为 5 代表识别到的是：右箭头

ID 值为 6 代表识别到的是：前进箭头

ID 值为 8 代表识别到的是：红心

ID 值为 10-19 代表识别到的是：数字 0-9

ID 值为 20-45 代表识别到的是：字母 A-Z

Function: vision_ctrl.get_marker_detection_info()

Return value:

- detection_info(list)

12.信息类（列表型数据）

获取识别到的行人或 S1 机器人信息，参数为 N，X，Y，W，H

注意：

物体信息的格式为：

第 1 项为识别到的物体数量 N，后续项以 4 个数据为一组，分别是第一个物体中心点在机器人视野中位置的横坐标 X、纵坐标 Y、宽度 W 和高度

H；第二个物体中心在机器人视野中位置的横坐标、纵坐标、宽度和高度；第三个……

Function: vision_ctrl.get_people_detection_info()

vision_ctrl.get_car_detection_info()

Return value:

- detection_info(list)

13.信息类（列表型数据）

获取识别到的姿势信息，参数为 N，ID，X，Y，W，H

注意：

1）姿势信息的格式为：

第 1 项为识别到的姿势数量 N，后续项以 5 个数据为一组，分别是第一个姿势的 ID 值，姿势中心点在机器人视野中的横坐标 X、纵坐标 Y，宽度 W 和高度 H；第二个姿势返回的 ID 值，中心点在机器人视野中的横坐标、纵坐标，宽度和高度；第三个……

2）ID 值释义：

ID 值为 4 对应姿势：正 V

ID 值为 5 对应姿势：倒 V 字

ID 值为 6 对应姿势：拍照手势

Function: vision_ctrl.get_pose_detection_info()

Return value:

- detection_info(list)

14.信息类（列表型数据）

获取识别到的线信息，参数为 N，Info，X，Y，θ，C

注意：

线信息的格式为：

第 1 项 N 为识别到的线上点的数量 10（固定值），第 2 项 Info 为返回的线类型（0 表示无线，1 表示一条线，2 表示 Y 字路口，3 表示十字路口），后续项以 4 个数据为一组：分别是线上由近及远的十个点的横坐标 X，纵坐标 Y，实际切线角 θ，曲率 C（取值范围 0-10，0 表示纯直线），一共 42 个数据。

Function: vision_ctrl.get_line_detection_info()

Return value:

- detection_info(list)

15.信息类（变量型数据）

获取当前场景的亮度信息，返回数值 0-10。数值越大，代表当前场景越亮。

Function: vision_ctrl.get_env_brightness()

Return value:

- brightness_value(int)

16.信息类（列表型数据）

获取准星位置信息，参数为 X，Y（准星跟随标签移动示例很好）

Function: media_ctrl.get_sight_bead_position()

Return value:

- sight_bead_position(list)

七、装甲板

1.设置装甲板灵敏度。数值越大，装甲板感应灵敏度越高。硬物敲击时建议灵敏度设为 6，指关节叩击时设为 8。灵敏度设置只在实验室环境中生效，对战中装甲板的灵敏度都会恢复为默认值。

Function: armor_ctrl.set_hit_sensitivity(value)

Parameters:

- value(int): [0, 10]

2.事件类(中断处理程序)

当检测到指定位置的装甲板受到攻击时，运行本模块内的程序

Function: def armor_hit_detection_all(msg)

def armor_hit_detection_bottom_right(msg)

def armor_hit_detection_bottom_left(msg)

def armor_hit_detection_bottom_front(msg)

def armor_hit_detection_bottom_back(msg)

def armor_hit_detection_top_right(msg)

def armor_hit_detection_top_left(msg)

Type: Event callback

3.信息类（变量型数据）

获取最近受到攻击的装甲板信息，ID 值反馈出装甲板位置，通过时间戳计算可以获知受攻击的时间点

注意：

返回的 ID 值释义：

ID 值为 1 对应受到攻击的装甲板在：底盘后侧

ID 值为 2 对应受到攻击的装甲板在：底盘前侧

ID 值为 3 对应受到攻击的装甲板在：底盘左侧

ID 值为 4 对应受到攻击的装甲板在：底盘右侧

ID 值为 5 对应受到攻击的装甲板在：云台左侧

ID 值为 6 对应受到攻击的装甲板在：云台右侧

Python API：

Function: armor_ctrl.get_last_hit_index()

Return value:

- index(int)

Function: armor_ctrl.get_last_hit_time()

Return value:

- time(float)

4.布尔型

持续检测指定装甲板是否受到攻击，被攻击会返回“真”，否则返回“假”

Function: armor_ctrl.check_condition(condition_enum)

Parameters:

- condition_enum(enum):
  - rm_define.cond_armor_hit
  - rm_define.cond_armor_bottom_front_hit
  - rm_define.cond_armor_bottom_back_hit
  - rm_define.cond_armor_bottom_left_hit
  - rm_define.cond_armor_bottom_right_hit
  - rm_define.cond_armor_top_left_hit
  - rm_define.cond_armor_top_right_hit

5.待指定位置的装甲板受到攻击后才会执行下一条指令，否则持续等待(执行类、阻塞型)

Function: armor_ctrl.cond_wait(condition_enum)

Parameters:

- condition_enum(enum):
  - rm_define.cond_armor_hit
  - rm_define.cond_armor_bottom_front_hit
  - rm_define.cond_armor_bottom_back_hit

- rm_define.cond_armor_bottom_left_hit
- rm_define.cond_armor_bottom_right_hit
- rm_define.cond_armor_top_left_hit
- rm_define.cond_armor_top_right_hit

八、移动设备

1.获取移动设备当前角度值

注意：

1）RoboMaster S1 的默认运动模式是“云台跟随底盘模式”，所以如果想单独控制云台绕航向轴旋转，需要设置为“自由模式”。

2）移动设备是指手机、平板等。

3）移动设备姿态角范围是 -180~180°，

对航向轴来说：向右为正（0-180）

对俯仰轴来说：向上为正（0-180）

对翻滚轴来说：右下为正（0-180）

Python API：

Function: mobile_ctrl.get_attitude(attitude_enum)

Parameters:

- attitude_enum(enum):
  - rm_define.mobile_atti_pitch
  - rm_define.mobile_atti_roll

- rm_define.mobile_atti_yaw

2.获取移动设备加速度单元的测量值

注意：

(1）速度变化越快，获取到的加速度测量值越大。

(2）移动设备是指手机、平板等。

Python API：

Function: mobile_ctrl.get_accel(axis_enum)

Parameters:

- axis_enum(enum):
  - rm_define.mobile_accel_x
  - rm_define.mobile_accel_y
  - rm_define.mobile_accel_z

九、多媒体

1.播放音效的同时，立刻执行下一条指令(执行类、非阻塞型)

Function: media_ctrl.play_sound(sound_enum, wait_complete_flag=False)

Parameters:

- sound_enum(enum):
  - rm_define.media_sound_attacked #被击中

- rm_define.media_sound_shoot　　#射击
- rm_define.media_sound_scanning#扫描中
- rm_define.media_recognize_success#识别成功
- rm_define.media_gimbal_rotate#云台旋转
- rm_define.media_sound_count_down#倒计时

2.音效播放完毕后才会执行下一条指令(执行类、阻塞型)

Function: media_ctrl.play_sound(sound_enum, wait_complete_flag=True)

Parameters:

- sound_enum(enum):
  - rm_define.media_sound_attacked
  - rm_define.media_sound_shoot
  - rm_define.media_sound_scanning
  - rm_define.media_recognize_success
  - rm_define.media_gimbal_rotate
  - rm_define.media_count_down

3.拍照。响起快门声的同时拍摄一张照片，照片会在相册中出现。

注意：

请提前在机器人中插入一张剩余用量在 2GB 以上的 SD 卡，无 SD 卡时拍照指令不生效。

Python API：

Function: media_ctrl.capture()

4.开始或结束视频录制，结束后会在 APP 相册和 SD 卡中生成一段视频。

Function: media_ctrl.record(enable_enum)

Parameters:

- enable_enum(enum):
  - 1
  - 0

十、控制语句

1.等待指定秒数，然后再执行下一条指令

Function: time.sleep(t)

Parameters:

- t(float): [0, 3600]s

2.重复 10 次,重复

Python 基本语法：for count in range(10)：、while True:

3.如果/否则也是 Python 基本语法

if……:

pass

else:

pass

4.停止正在运行的所有程序

Function: rmexit()

十一、运算符

加减乘除、取余(%)以及逻辑运算(==/!=/>=/and/or/not)为 python 基本语法

利用了 math 等模块的一些函数和常数，如下：

0.math 定义的两个常数：math.pi;math.e

1.random.randint(1,10) #随机取 1 到 10 之间的整数，包括 1 和 10

注：需要 import random

2.rmround(3.5) #将 3.5 四舍五入

3.abs() #取绝对值

4.math.floor()、math.ceil() #向下、上取整

5.math.sin();math.cos();math.tan() #要求输入弧度值，如用度需换算为

math.sin(x/180*math.pi)

6.math.asin();math.acos();math.atan() #要求输入弧度值，如用度需换算为

math.asin(x)/math.pi*180

7.math.sqtr() #平方根

8.math.exp() #e^

9.math.pow(10,2) #10^2

10.math.log(2,math.e) #ln2

11.math.log(3,10) #log3

十二、数据对象

1.基本变量和列表

变量的使用同 python 语法，列表为大疆自定义的类（下标从 1 开始，而不是从 0），但用法基本同原列表

示例：

```
variable_i = 0                          #定义变量
list_list01 = RmList()                  #定义一个空列表
def start():
    global variable_i                   #声明使用全局变量
    global list_list01
    global pid_pid01
    variable_i = 3                      #基本变量赋值
    variable_i = variable_i + 1         #基本变量自增减
    list_list01=RmList()                #将 list_list01 列表设为空列表
    list_list01.append(1)               #将 1 添加到列表末尾
    list_list01.pop(2)                  #删除列表第 2 项
    list_list01.clear()                 #删除列表全部项
    list_list01.pop()                   #删除列表末尾项
    list_list01.insert(1, 2)            #在第一项前插入 2
    list_list01[1]                      #列表的第一项
```

```
list_list01.index(2)              #列表中第一个 2 的索引
list_list01[1] = 9                #将列表第一项替换为 9
len(list_list01)                  #列表的长度
if (2 in (list_list01)):          #  “in”成员判定
    pass
```

2.PID 对象

示例：

```
variable_i = 0
pid_pid01 = rm_ctrl.PIDCtrl()          #定义一个 PID 控制器 pid_pid01
def start():
    global variable_i
    global pid_pid01
    pid_pid01.set_error(5)             #设置 PID 控制器 pid_pid01 的误差为 5
    #设置 PID 控制器 pid_pid01 的比例、积分、微分参数 Kp、Ki、Kd 为 1、2、3
    pid_pid01.set_ctrl_params(1,2,3) #参数为 float 类型
    #将变量 variable_i 设为 PID 控制器的输出
    variable_i = pid_pid01.get_output()
```