ROS interface for mobile robots (external)

Based on customer requirements and to facilitate education and research customers to integrate our chassis, our company provides the internal ROS interface and calling method of the chassis.

Note: The ROS interface is designed for internal use by our company. Its interface path, data structure, calling method, behavior, etc. may change with the changes in the chassis internal software. When you plan to upgrade the chassis software, please confirm the changes in the internal interface you are using with our company and modify your program in time according to the changes.

Note: Some ROS interfaces bypass the chassis' safety protection function. Improper use may threaten personal and property safety. Please ensure safety monitoring. If necessary, please consult our company.

You can connect to the chassis' ROS interface through the debug Ethernet interface or WiFi. For stability considerations, we recommend that you use the debug Ethernet interface to call the chassis' ROS interface.

Table of contents

ROS version

Currently, the ROS version used by the chassis is ROS noetic and the operating system is Ubuntu 20.04. It is recommended that you use the same ROS version and operating system version. In the following text, we assume that you are also using Ubuntu 20.04.

Connect to chassis via debug Ethernet interface

IP address configuration

First, you need to connect your computer's wired Ethernet interface to the chassis' debug Ethernet interface. You can refer to the user manual to complete this operation.

You will then need to configure a static IP address for your computer. The IP address is between 169.254.128.20-169.254.128.49, and the subnet mask is 255.255.255.0.

After the IP address configuration is complete, you can use the ping tool to verify whether the network configuration is effective. Open Terminal on your computer and enter ping 169.254.128.2. If it returns 64 bytes from 169.254.128.2: icmp_seq=1 ttl=64 time=0.203 ms or other similar information, the IP address configuration is successful.

Environment variable configuration

Temporary validity

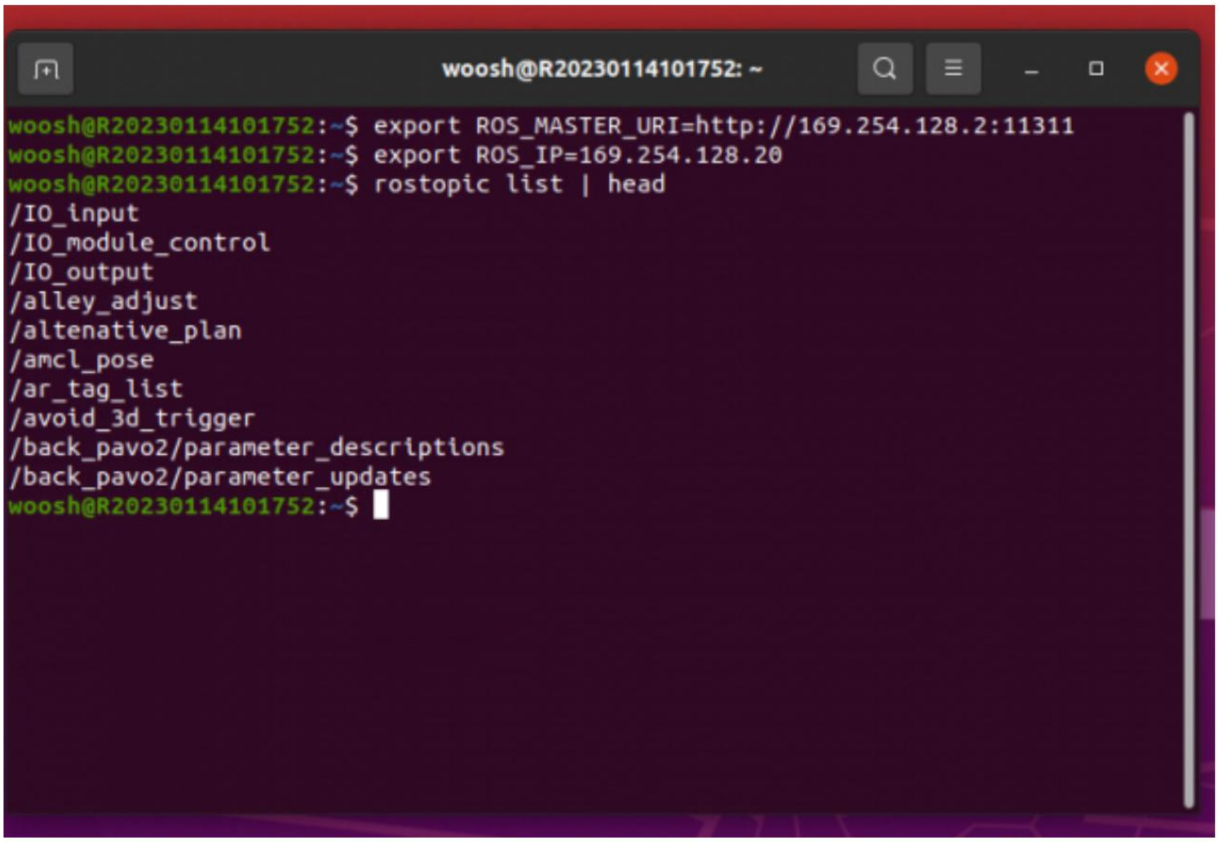
Specify ROS_MASTER_URI and ROS_IP in Terminal. Here we assume your IP address is 169.254.128.20

Enter in Terminal (valid for the current Terminal)

export ROS_MASTER_URI=http://169.254.128.2:11311

export ROS_IP=169.254.128.20

At this point, all ROS programs running in the current terminal will use 169.254.128.2 as the ROS master node. As shown in the following figure:



```
woosh@R20230114101752:~$ export ROS_MASTER_URI=http://169.254.128.2:11311
woosh@R20230114101752:~$ export ROS_IP=169.254.128.20
woosh@R20230114101752:~$ rostopic list | head
/IO_input
/IO_module_control
/IO_output
/alley_adjust
/altenative_plan
/amcl_pose
/ar_tag_list
/avoid_3d_trigger
/back_pavo2/parameter_descriptions
/back_pavo2/parameter_updates
woosh@R20230114101752:~$
```

Permanent

Add the following to the end of .bashrc: (assuming your IP address is 169.254.128.20)

export ROS_MASTER_URI=http://169.254.128.2:11311

export ROS_IP=169.254.128.20

Common (ROS) interface description

Subscribe to data (topic)

Get robot status

Function description: Get the robot's current task status information, and publish it only when there is a change

Interface name (topic): robot_status

Message type: woosh_msgs/RobotStatus

Specific parameter description:

Parameter name type		meaning	whether Required	about bundle
task_id	uint64 Task ID		yes	
task_type	int32 #Task type, 0: undefined, 1: picking, 2: parking, 3: charging, 4: transporting		yes	
task_state	int8	#Task status, 0: undefined, 3: executing, 4: paused, 5: waiting for action, 7: completed 8: Cancelled, 9: Failed	yes	
action_type int8		#Action type 0: undefined, 1: navigation, 2: single-step control, 3: secondary positioning entry, 4: secondary Positioning exit, 5: transport action, 6: waiting, 7: charging	yes	
action_state int8		#Action status 0: Undefined, 1: Executing, 2: Warning, 3: Cancel, 4: Completed, 5: Failed Defeat, 10: Suspension, 11: Control	yes	
robot_state int8		Robot status 0: Undefined, 1: Uninitialized, 2: Idle, 3: Parking, 4: On a mission, 5: Warning, 6: Abnormal, 8: Charging, 9: Composition	yes	
robot_mode int8 Robot mode 0: Undefined, 1: Automatic, 2: Manual, 3: Maintenance			yes	
work_mode int8 #Robot working mode 0: undefined, 1: deployment mode, 2: task mode, 3: scheduling mode				
wait_id	int32 #Action waiting ID			
dest	string #target point			
msg	string #message			
time	int32 #Last updated time (s)			

Robot positioning information acquisition

Function description: Publish plane coordinate information

Interface name (topic): movebase_pose2d

Message type: geometry_msgs/Pose2D

Specific parameter description:

Parameter name	type	meaning	Is the constraint required?
x	float64	The x-axis coordinate of the Cartesian coordinate system	yes
y	float64	The y-axis coordinate of the Cartesian coordinate system	yes
theta	float64	Cartesian coordinate system z-axis coordinate	yes

Odometer information

Function description: Publish machine odometer information

Interface name: odometer

Message type: std_msgs/Float64

Specific parameter description:

Parameter name	type	meaning	Is the constraint required?
data	float64	Odometer (unit: meter)	Yes

Power Information

Function description: Publish the robot's power information

Interface name: /battery

Message type: woosh_msgs/Battery

Specific parameter description:

parameter name	type	Meaning	Is it required?	constraint
parameter name	type	Meaning	Is it required?	constraint
header	Header		yes	
submodule	uint8	submodule	yes	
batteryPercentage	uint8	The battery power percentage is		
chargeMode	uint8	Charging mode	yes	0 (default) - Manual charging 1- Automatic charging
chargeStatus	uint8	Charging status	yes	0 - Not charging 1-Charging
batteryVoltage	float32	battery voltage	yes	
batteryCurrent	float32	battery current	yes	
timeRemain	float32	remaining time	yes	
tempMax	int8	maximum temperature	yes	
tempMin	int8	minimum temperature	yes	
capacityRemain	float32	remaining capacity	yes	
capacityFull	float32	full capacity	yes	
capacityDesign	float32	design capacity	yes	
chargeCycle	uint16	number of loops	yes	
batteryCycle	uint16	Battery life	yes	
bmsStatus	uint32	bms status	no	
cellVoltageMax	float32	maximum cell voltage	no	
cellVoltageMin	float32	minimum cell voltage	no	

Robot positioning information acquisition

Function description: Subscribe to robot SLAM positioning coordinate information.

Interface name (topic): /amcl_pose

Message type: geometry_msgs/PoseWithCovarianceStamped

Lower computer information

Function description: Publish the version and parameter information of the lower computer

Interface name: /information

Message type: woosh_msgs/Information

Specific parameter description:

parameter name	type	Meaning	Is the constraint required?	
header	std_msgs/Header	Frame Header	yes	
hardwareVersion	string	The hardware version number	yes	
softwareVersion	string	The software version number	yes	
IAPSoftwareVersion string		The IAP software version number	yes	
movebaseType	string	Chassis type	yes	
parameter	woosh_msgs/Parameter	Chassis parameters are	yes	

Battery BMS information

Function description: Release battery BMS information

Interface name: /driver_base/bms

Message type: woosh_msgs/BMS

Specific parameter description:

parameter name	type	Meaning	Is the constraint required?	
header	std_msgs/Header	frame header	yes	
BMSStatus	uint32	BMS Status	No	
current	float32	Current	yes	
voltageMax	float32	Maximum cell voltage	No	
voltageMin	float32	Minimum cell voltage	No	
voltageTotal	float32	The total battery voltage	yes	
temperatureMax float32		The maximum temperature is	yes	
temperatureMin float32		Minimum temperature	No	
chargeCycle	uint16	The number of cycles	yes	
capacityResidue float32		The remaining capacity	yes	
capacityTotal	float32	total capacity	yes	
switchStatus	uint8	Switch Status	No	

Lower computer parameter information

Function description: Publish the lower machine parameter information

Interface name: /driver_base/parameter

Message type: woosh_msgs/Parameter

Specific parameter description:

parameter name	type	meaning	Is the constraint required?	
header	std_msgs/Header	frame header	yes	
wheelPerimeter	float32	Wheel circumference	yes	
wheelDistant	float32	Wheel spacing	yes	
ratioLeft	float32	Left motor reduction ratio	yes	
ratioRight	float32	Right motor reduction ratio	yes	
pulsePerCircle	float32	Number of pulses per revolution	yes	
maxRPM	float32	Revolutions per minute	yes	
minSpeed	float32	Minimum speed	yes	
maxSpeed	float32	Maximum speed	yes	
maxLinear	float32	Speed control maximum line speed	yes	
maxAngular	float32	The maximum angular velocity of speed control	yes	
joystickMaxLinear	float32	The maximum linear speed of the joystick	yes	
joystickMaxAngular	float32	The maximum angular velocity of the joystick	yes	

Speed information

Function description: Publish the robot's speed information

Interface name: odom_twist

Message type: geometry_msgs/Twist

Specific parameter description:

parameter name	type	Meaning	Is the constraint required?	
linear	geometry_msgs/Vector3	Linear velocity is	yes	
angular	geometry_msgs/Vector3	The angular velocity is	yes	

status code

Function description: Publish the status code of the robot

Interface name: status_code

Message type: std_msgs/UInt64

Specific parameter description:

Parameter name	type	meaning	is it required?	constraint
data	The uint64 exception code is			See Robot Status Code and Exception Handling Mechanism for details.

speed control

Note: Speed control will bypass the chassis' obstacle avoidance, collision detection, speed limit, state limit and other safety functions. Please make sure before calling Make sure that the area around the robot is clear and that someone is watching over it. The caregiver needs to closely monitor the robot while ensuring their own safety. status, and intervene immediately when necessary to stop the robot's movement.

Function description: Subscription speed control instructions (without smoothing)

Interface name: /base_cmd_vel

Message type: geometry_msgs/Twist

Specific parameter description:

parameter name	type	Meaning	Is the constraint required?	
linear	geometry_msgs/Vector3 Linear velocity is			
angular	geometry_msgs/Vector3 The angular velocity is			

Function description: Subscribed speed control instructions (with deceleration smoothing processing)

Interface name: /smooth_cmd_vel

Message type: geometry_msgs/Twist

Specific parameter description:

parameter name	type	Meaning	Is the constraint required?	
linear	geometry_msgs/Vector3 Linear velocity is	yes		
angular	geometry_msgs/Vector3 The angular velocity is	yes		

Light strip color control

Function description: Subscribed light strip color control instructions

Interface name: rgbled, rgbled_shelf

Message type: woosh_msgs/LED

Specific parameter description:

Parameter name type meaning is it required?				constraint
Parameter name type meaning is it required?				constraint
submodule	uint8	submodule	no	0-Select All 1~8-Module 1~8
urgency	uint8	Maintenance status	No	0x00-Not selected 0xFF-Maintenance
abnormal	uint8	Fault status	No	0x00-Not selected 0xFF - Failure
normal	uint8	Normal status	no	0x00-Not selected 0x42 - Low Battery 0x60-Follow 0x80 - Forward 0x81-Turn right 0x82 - Turn Left 0x83 - Boot from Standstill 0x84-Stop waiting 0x85-Traffic control is suspended 0xA0 - Warning 0xA1-Task Execution 0xA2-Task Pause 0xA3 - Idle 0xA4-Initialization 0xA5-Offline 0xF0-Picking task execution (Guanbang) 0xF1-Picking task completed (Guanbang) 0xFF - Normal
color	uint32	24-bit true color	no	

Lidar data

Function description: Subscribe to lidar data (data after fusion of multiple lidars) for indoor positioning and information fusion.

Interface name (topic): /scan

Message type: sensor_msgs::LaserScan

IMU data acquisition

Function description: Subscribe to the raw data of IMU for indoor positioning and information fusion.

Interface name (topic): /imu/data_raw

Message type: sensor_msgs/Imu

odom data acquisition

Function description: Subscribe to odom's posture information, raw data (without IMU fusion), for indoor positioning and information fusion.

Interface name (topic): /odom_raw

Message Type: nav_msgs/Odometry

Function description: Subscribe to odom's raw data (fused with IMU) for indoor positioning and information fusion.

Interface name (topic): /odom

Message Type: nav_msgs/Odometry

Specific parameter description:

parameter name	type	Meaning	Is the constraint required?	
header	std_msgs/Header	The frame header is	yes	
child_frame_id string			yes	
pose	geometry_msgs/PoseWithCovariance	coordinates	yes	
twist	geometry_msgs/TwistWithCovariance	speed	yes	

RGB camera data acquisition

Function description: Subscribe to the raw data of the RGB camera

Interface name (topic): /camera_1/color/image_raw

Message type: sensor_msgs/Image

Camera depth image data acquisition

Function description: Subscribe to the raw data of the camera depth image

Interface name (topic): /camera_1/depth/image_raw

Message type: sensor_msgs/Image

Camera 3D point cloud data acquisition

Function description: Subscribe to the raw data of the camera depth image

Interface name (topic): /camera_1/depth/points

Message type: sensor_msgs/PointCloud2

Obstacle avoidance 3D point cloud data acquisition

Function description: Subscribe to camera stereo obstacle point cloud data (remove ground point cloud information)

Interface name (topic): /camera_1/depth/cloud_without_planes

Message type: sensor_msgs/PointCloud2

Obstacle avoidance 3D point cloud data acquisition

Function description: Subscribe to camera stereo obstacle point cloud data (remove ground point cloud information)

Interface name (topic): /camera_1/depth/cloud_without_planes

Message type: sensor_msgs/PointCloud2

Modify the local (local_costmap) model (footprint) of the robot

Function description: Set the size of the local footprint. Support changing the size and shape of the vehicle's footprint under the local costmap during navigation. This change will cause the local_costmap to change (expand or shrink).

This affects the effect of the car's local path planning. It is recommended to modify both the global and local models at the same time.

Interface name: /move_base/local_costmap/set_footprint

Message type: geometry_msgs::Polygon

Specific parameter description:

parameter name	**type**	**meaning**	**Is the constraint required?**	
points	geometry_msgs/Point32[]	The polygon connected from the first point to the last point, that is, the footprint is	yes	

Modify the robot's global (global_costmap) model (footprint)

Function description: Set the size of the global footprint. Support changing the size and shape of the car's footprint under the global costmap during navigation. This change will affect the global_costmap as well (expand or shrink).

This affects the effect of the car's global path planning. It is recommended to modify both the global and local models at the same time.

Interface name: /move_base/global_costmap/set_footprint

Message type: geometry_msgs::Polygon

Specific parameter description:

parameter name	**type**	**meaning**	**Is the constraint required?**	
points	geometry_msgs/Point32[]	The polygon connected from the first point to the last point, that is, the footprint is	yes	

Calling Example

For example, the model size of footprint is as follows:

Footprint: [[-0.400, 0.253], [-0.353, 0.3], [0.353, 0.3], [0.4, 0.253], [0.4, -0.253], [0.353, -0.3],
[-0.353, -0.3], [-0.4, -0.253]]

footprint_padding: 0.1 expansion 0.1

```
#include <ros/ros.h>
#include <geometry_msgs/Polygon.h>
#include <geometry_msgs/Point.h>

int main(int argc, char** argv)
{
   ros::init(argc,argv,"test_set_footprint_node");
   ros::NodeHandle nh;

   ros::Publisher local_footprint_pub,global_footprint_pub;
    local_footprint_pub = nh.advertise< geometry_msgs::Polygon > ("/move_base/
local_costmap/set_footprint", 1);
   global_footprint_pub = nh.advertise< geometry_msgs::Polygon > ("/move_base/
global_costmap/set_footprint", 1);
   //The polygon formed by connecting 8 points is the model of the car
     std::vector< geometry_msgs::Point32 > pts;
     geometry_msgs::Point32 point32;
     point32.x = -0.400;
     point32.y = 0.253;
     point32.z = 0.0;
     pts.push_back(point32); //Starting point
     point32.x = -0.353;
     point32.y = 0.3;
     point32.z = 0.0;
     pts.push_back(point32); //The second point
     point32.x = 0.353;
     point32.y = 0.3;
     point32.z = 0.0;
     pts.push_back(point32); //The third point
     point32.x = 0.400;
     point32.y = 0.253;
     point32.z = 0.0;
     pts.push_back(point32); //The fourth point
     point32.x = 0.400;
     point32.y = -0.253;
     point32.z = 0.0;
    pts.push_back(point32); //The fifth point
    point32.x = 0.353;
    point32.y = -0.3;
    point32.z = 0.0;
    pts.push_back(point32); //The sixth point
    point32.x = -0.353;
    point32.y = -0.3;
    point32.z = 0.0;
    pts.push_back(point32); //The seventh point
    point32.x = -0.4;
    point32.y = -0.253;
    point32.z = 0.0;
    pts.push_back(point32); //The eighth point

    sleep(1);
  // Connect into polygons, i.e. footprint
    geometry_msgs::Polygon polygon;
    for (int i = 0; i < pts.size(); i++)
    {
        polygon.points.push_back(pts[i]);
    }
    //Modify the local footprint
    local_footprint_pub.publish(polygon);
  //Modify the global footprint
    global_footprint_pub.publish(polygon);
    while(ros::ok())
    {
    }
  return 0;
}
```

Service Communication Interface

Task Execution

Functional description: Execute task interface

Interface name: exec_task

Request message type: woosh_msgs::ExecTask

Request specific parameter description:

parameter name	type	meaning	whether Required	about bundle
task_exect	uint8	#Task execution request, 1: execute, 2: pause, 3: continue, 4: Cancel, 6: Wait for interruption	yes	
task_id	int64	#Task ID	yes	
task_type	uint8	#Task type, 1: Picking 2: Parking 3: Charging 4: Transporting		
direction	uint8	#Action direction, 0: Undefined 1: Loading 2: Unloading, not loading Fill in 0 for the material cutting task, optional	yes	
task_type_no uint32		#Type combination, default 0, custom, optional	yes	
mark_no	string	#Target point (storage location) number. If you fill in the number, the task route The last item of the task path can be left blank. points, must be consistent with this storage point		
poses	geometry_msgs/PoseStamped[]	#Task path, if it is a point (target point), it means The system plans by itself and reaches this target point eventually. Optional		
custom	byte	# Custom fields, vary by project, optional		

Response parameter description:

Parameter name type meaning			Is it necessary fill	constraint
success	bool	Responding to a request state	yes	true - indicates the request was successful false - indicates the request failed
message string		Comment Response information	no	
statuscode uint64 #status is			yes	1: Uninitialized, 2: Idle, 3: Parking, 4: Tasking, 5: Warning, 6: Abnormal, 8: Charging, 9: Composing

Start/stop positioning module

Function description: Enable positioning function

Interface name: /start_localization

Request message type: std_srvs::SetBool

Request specific parameter description:

Parameter name type		meaning	Required?	constraint
data	bool Start/stop positioning module		yes	true-indicates starting the positioning module false-indicates closing the positioning module

Response parameter description:

Parameter name type meaning is it required?				constraint
success bool The response request status is			yes	true - indicates the request was successful false - indicates the request failed
message string Remarks response information No			no	

Start/stop costmap update

Function description: Enable cost map data update function

Interface name: /move_base/enable_costmaps

Request message type: std_srvs::SetBool

Request specific parameter description:

parameter name	type	meaning	Required?	constraint
data	bool	Start/stop costmap data update	yes	true-indicates starting the costmap update, which can update the obstacle data to the costmap false-indicates turning off costmap updates and not updating obstacle data to the costmap

Response parameter description:

Parameter name type meaning is it required?				constraint
success bool The response request status is			yes	true - indicates the request was successful false - indicates the request failed
message string Remarks response information No			no	

Get map data

Function description: Request current map data information from map_server

Interface name: aic_map_server/get_map

Message Type: nav_msgs::GetMap

Response parameter description:

parameter name	type	meaning	Is the constraint required?	
map	nav_msgs/OccupancyGrid	The map data information obtained by the client is		

Change Map

Function description: Replace the map data information sent

Interface name: set_map

Message type: nav_msgs::SetMap

Request specific parameter description:

parameter name	type	meaning	Required?	constraint
map	nav_msgs/OccupancyGrid	Map data information sent by the client	yes	
initial_pose	geometry_msgs/PoseWithCovarianceStamped	The position information of the robot's initial point (current point) sent by the client	yes	If it is the initialization point, the general robot posture is (0,0,0) If it is a multi-map navigation, the value is the current robot's position information relative to this map

Response parameter description:

Parameter name	type	meaning	is it required?	constraint
success	bool	The response request status is		true - indicates that the request was successful. false - indicates that the request failed

Planning the global path

Function description: Request global path planning information

Interface name: /move_base/make_plan

Request message type: nav_msgs/GetPlan

Specific parameter description:

parameter name	type	meaning	whether Required	constraint
start	geometry_msgs/PoseStamped	The starting point of the path	yes	
goal	geometry_msgs/PoseStamped	The target point of the path	yes	
tolerance	float32	The offset of the target point. When the target point is blocked, It is possible to allow deviation to avoid path planning failure	no	Note: Do not deviate too much More will lead to greater errors

Response message type: nav_msgs/Path

Specific parameter description:

parameter name	type	meaning	Is the constraint required?	
header	std_msgs/Header	Contains timestamp and coordinate system information		
poses	geometry_msgs/PoseStamped[]	The path coordinate information of the navigation to the target point is		

Clear obstacles

Function description: Request global path planning information

Interface name: /move_base/clear_costmaps

Request message type: std_srvs::Empty

Request specific parameter description:

Parameter name	type	meaning	whether it is required	constraint
none				

Response parameter description:

Parameter name	type	meaning	whether it is required	constraint
none				

Action API

Navigation target point

Function description: Send the target point or path to be navigated

Interface name: /move_base/goal

Message type: woosh_msgs/MoveBaseActionGoal

Specific parameter description:

parameter name	type	meaning	whether Required	constraint
poses	geometry_msgs/PoseStamped[]	Navigate to the destination Radius coordinate information	no	
target_pose	geometry_msgs/PoseStamped	Navigate to the target point Label Information	yes	Regardless of whether the path is sent, this parameter target_pose is required

High-precision speed combination control (suitable for pure motion control and high-precision walking)

Function description: Subscribed speed control instructions (with deceleration smoothing processing)

Interface name: /cmd_vel_control

Message type: woosh_msgs::StepControlGoal

Specific parameter description:

parameter name	type	meaning	Is it necessary fill	constraint
parameter name	type	meaning	Is it necessary fill	constraint
mode	uint8	execution succeed Logo	yes	CANCEL EXCUTE - Execute PAUSE RESUM - Continue Note: In the case of multiple actions combined, you need to resend after pausing. The remaining action combinations
useAvoid	bool	Is obstacle avoidance	yes	false - no obstacle avoidance true-obstacle avoidance
StepControl[]	StepControl	step combination is	yes	
Result	bool	The execution result is	yes	
feedback	int8	The execution status is	yes	
percent	float	Completion percentage Compare	No	Unit: %

StepControl Parameters

executeMode uint8 Execution mode is					STRAIGHT-Go straight ROTATE
data	The float32 execution data is the maximum straight distance of 10m and the maximum rotation radius of 62.8318rad				
speed	float32 Maximum speed No Default linear speed 0.2m/s, angular speed 0.1rad/s				

Navigation mode switch

Function description: Switching of navigation mode, that is, making corresponding navigation motion strategies according to the sent mode or maximum running speed

Interface name: /navigation_mode/goal

Message type: woosh_msgs::NavigationModeGoal

Specific parameter description:

parameter name	type	meaning	whether Required	constraint
navigation_mode	uint8	Navigation mode	yes	0- indicates fuzzy avoidance navigation mode 1- indicates precise avoidance navigation mode 2- Indicates stop and wait navigation mode 3- Waiting for avoidance, wait for a certain period of time before executing avoidance
waiting_timeout	float32	After the waiting time has expired, perform an evasive maneuver. The unit is seconds float32	no	If the navigation mode is 3, the parameter value must be set to >= 0. It is considered as precise avoidance navigation mode
max_speed	maximum	navigation speed	No less than	or equal to 0 - indicates that the default speed of YAML is enabled