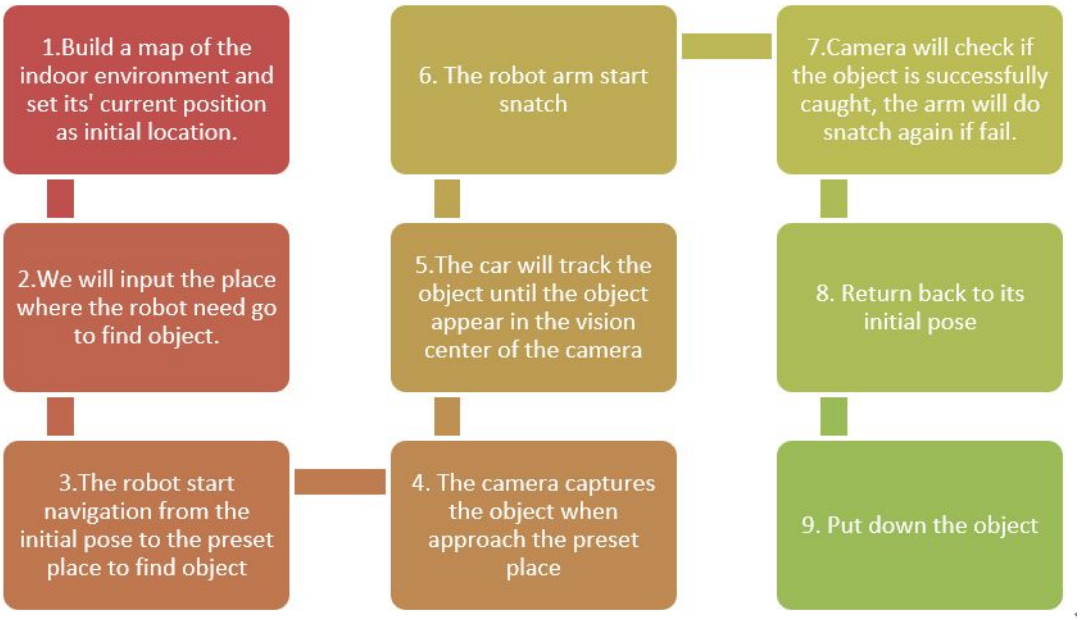
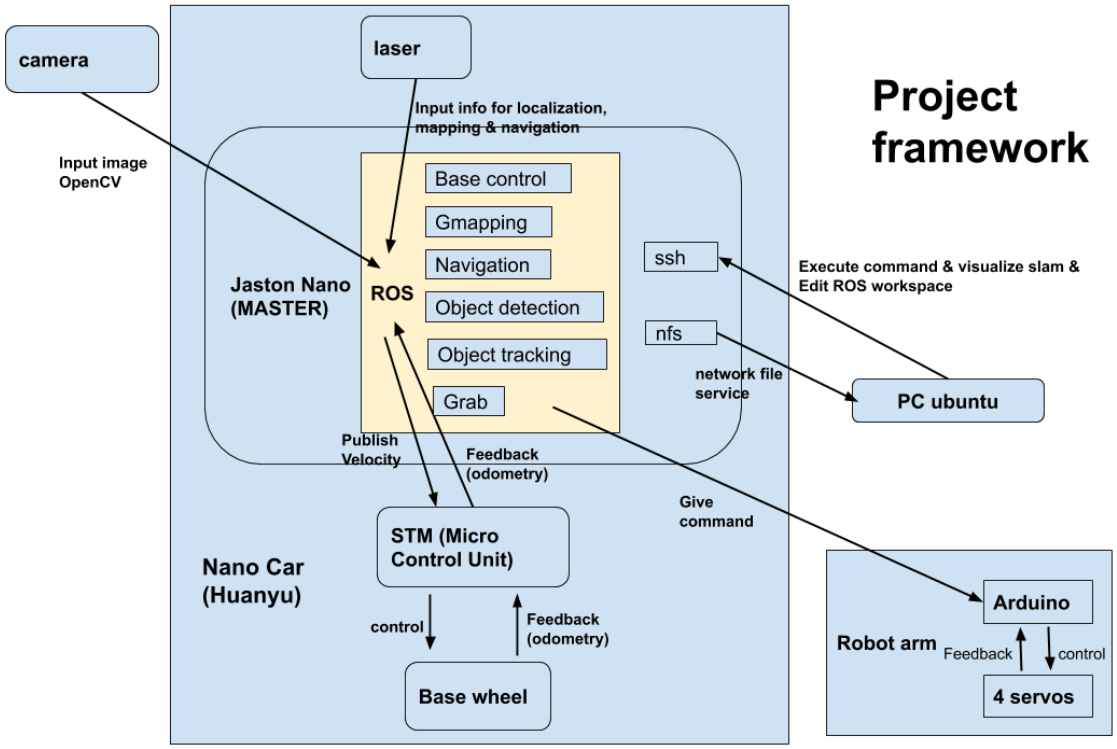
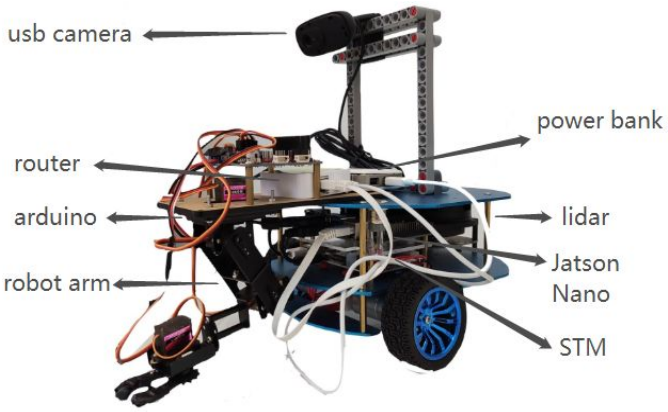
Final Year Project: Self-navigation robot for search and rescue

Workflow:





Robot structure + Hardware + Software



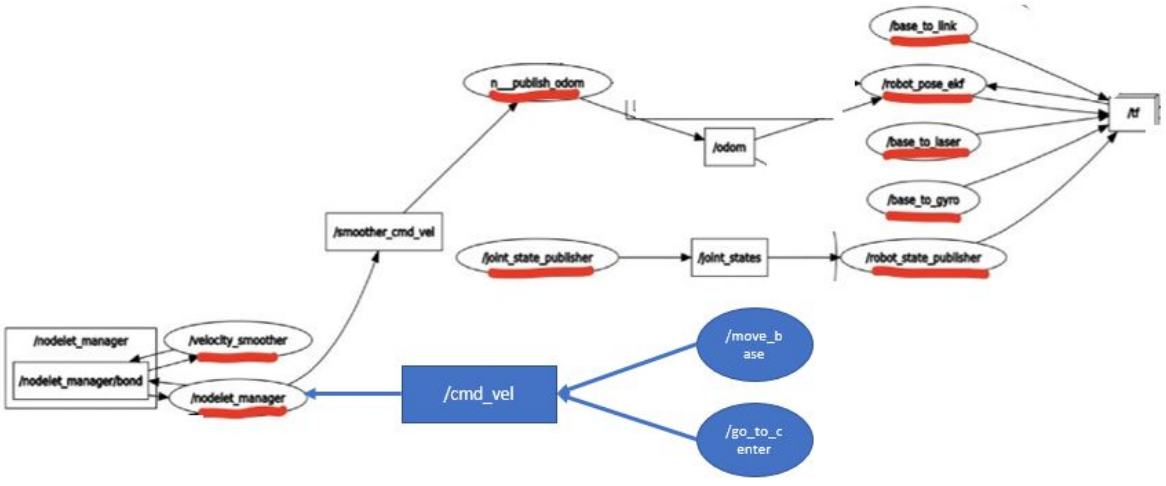
Hardware		
Item	**Provider**	**Function**
Jatson Nano Developer Kit	Huanyu-Hangzhou	Process data, main control center
Router	Huanyu-Hangzhou	Connect the Jetson Nano with PC and Internet
Lidar	Huanyu-Hangzhou	Scan
HiBot board	Huanyu-Hangzhou	Control board for Lidar and Motor
Camera	Logitech	Detect object
Arduino board	Taobao-1	Control board for robotic arm
Robotic arm + servo	Taobao-1	Grab object
PC	Self used PC	Remote control Jetson Nano, monitor task procedure

Software/tool/ Library	
Item	**Description**
Ubuntu 18.04	OS for Robot Operating system
Arduino	Platform to code robot arm
OpenCV	Image processing
Robot Operating System (ROS)	Platform to develop robot projects
Rviz	Visualization of images and topics

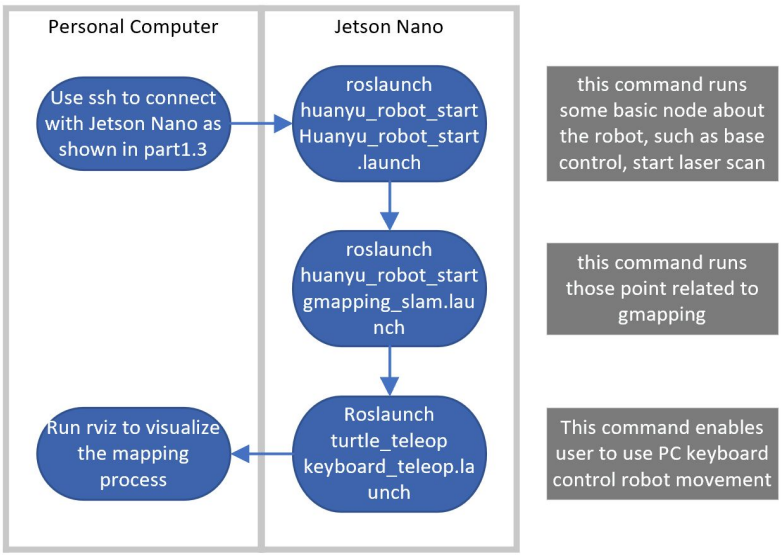
Functions

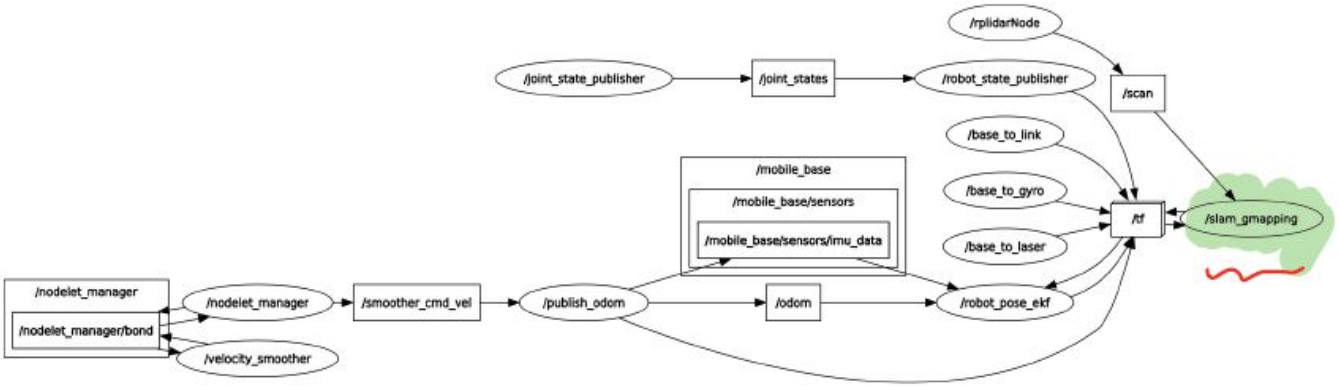
Function			
Number	Name of the function	Major Program provider	Description
1	Base control	Huanyu	Full code available and accessible inside Nano robot for base package control
2	Mapping (SLAM)	ROS	Open source : gmapping
3	Navigation (SLAM)	ROS	Open source package: move_base
4	Object detection	Ourselves	Detect the object and get the center location
5	Object tracking	Ourselves	Move Nano to an appropriate location for picking up the object
6	Grab	Ourselves	Pick up the object and put down the object (code in arduino)
7	Path point	Ourselves	Record points for searching target and navigate robot to return when grab is finish

1.Base control

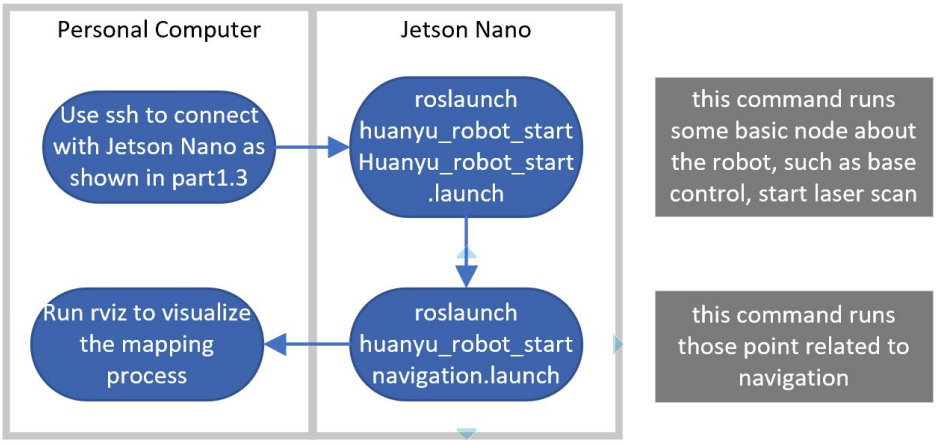


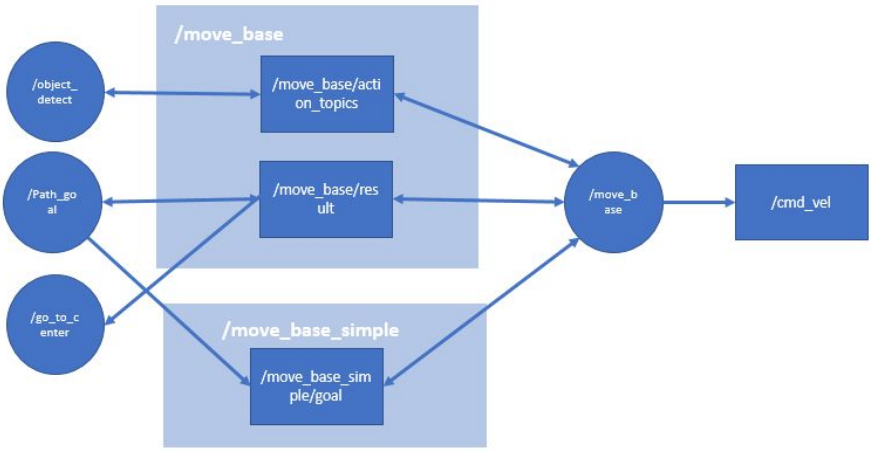
2.Mapping





3.Navigation

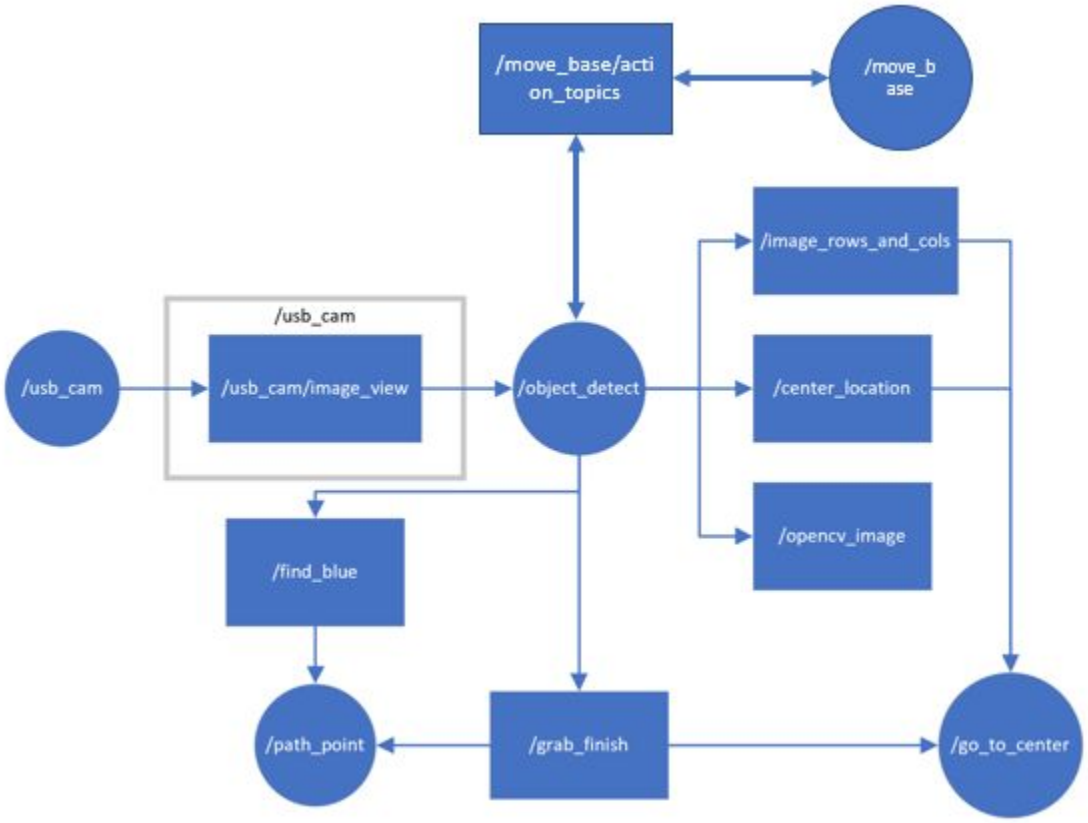




Node name: move_base (incomplete list)			
Topic/Service	Direction	Message type	Function
/move_base_simple/goal	Subscriber	geometry_msgs/PoseStamped	Get a goal for move_base to pursue in the map
/move_base/result	Publisher	move_base_msgs/MoveBaseActionResult	Indicate whether a navigation is finished
/cmd_vel	Publisher	geometry_msgs/Twist	Publish velocity to robot
SimpleActionClient<ActionSpec>::cancelGoal(Action achieved through move_base/action_topics)	Server	actionlib::SimpleActionClient	Cancel the goal that are currently running

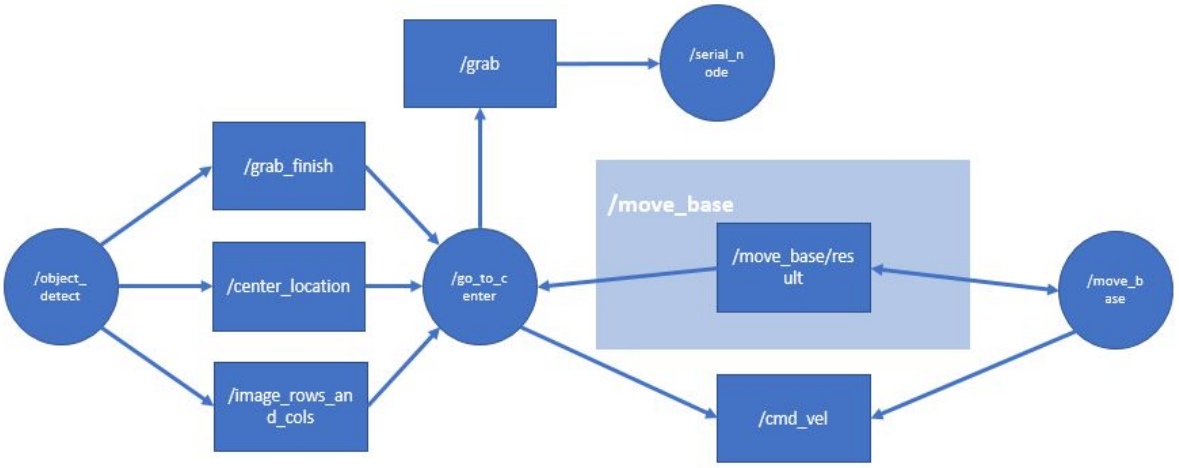
4. Obeject_detect

Node name: usb_cam		
Topic	Direction	Message type
/usb_cam/image_raw	Publisher	sensor_msgs/Image

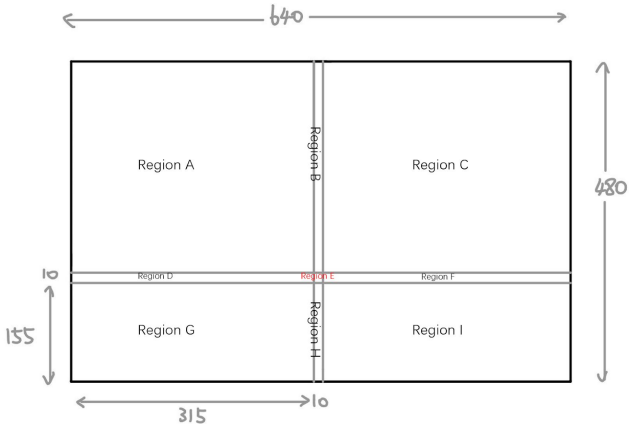


Node name: object_detect			
Topic/Action	Direction	Message type	Function
/usb_cam/image_raw	Subscriber	sensor_msgs/Image	get camera image
/opencv_image	Publisher	sensor_msgs/Image	visualize OpenCV processed image
/image_rows_and_cols	Publisher	geometry_msgs/Point	output image info
/center_location	Publisher	geometry_msgs/Point	reveal object center location
/find_blue	Publisher	geometry_msgs/Point	Indicate the object first appeared in the image
/grab_finish	Publisher	geometry_msgs/Point	Indicate the completion of grab task
SimpleActionClient<ActionSpec>::cancelGoal() (Action achieve through move_base/action_topics)	Client	actionlib::SimpleActionClient	Cancel the goal that are currently running

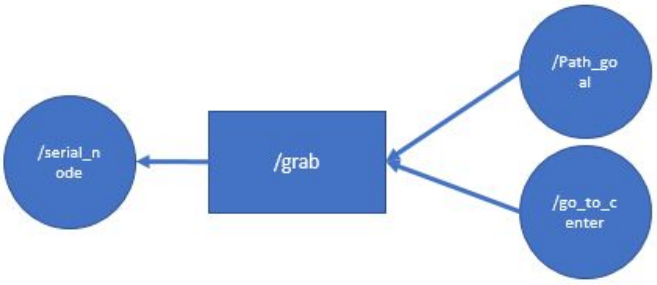
5.Object Tracking



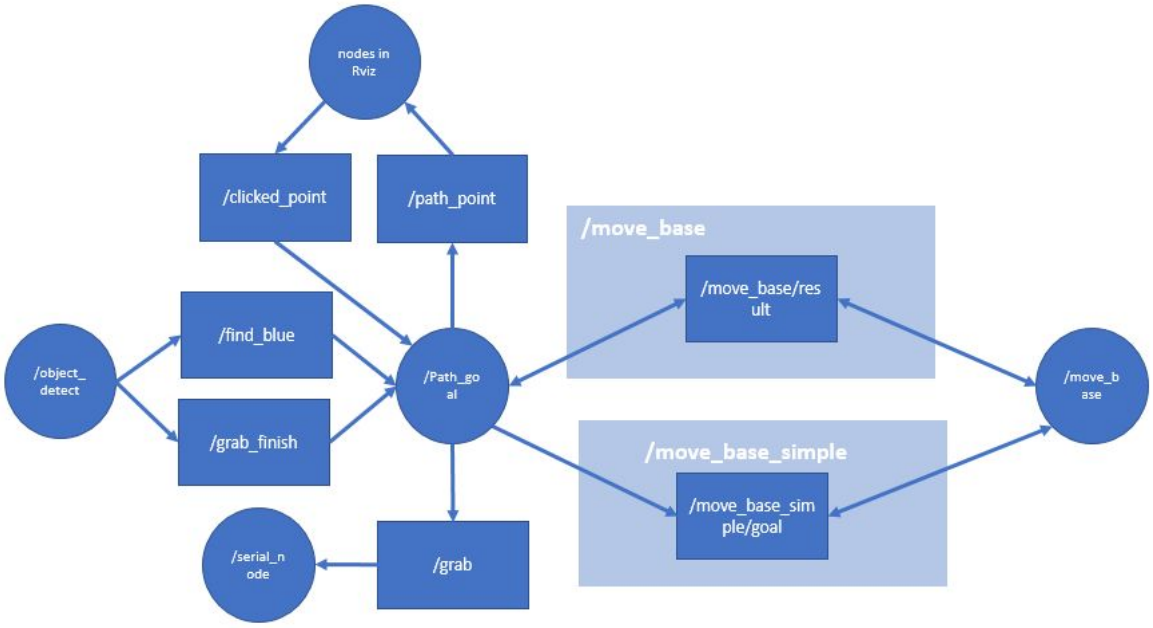
Node name: object_detect			
Topic/Action	Direction	Message type	Function
/move_base/result	Subscriber	move_base_msgs/MoveBaseActionResult	Check if navigation complete
/image_rows_and_cols	Subscriber	geometry_msgs/Point	get frame(image) info
/center_location	Subscriber	geometry_msgs/Point	get object center location
/grab_finish	Subscriber	geometry_msgs/Point	Check if grab task is finished
/cmd_vel	Publisher	geometry_msgs/Twist	controls the velocity of robot
/grab	Publisher	geometry_msgs/Point	Make the arm to pick up the target



6. Grab - Arduino Robot Arm

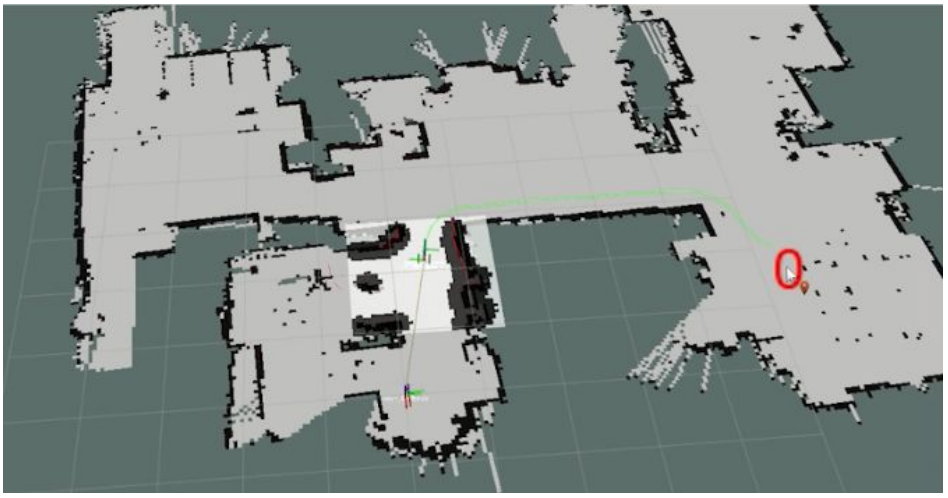


7.Path point

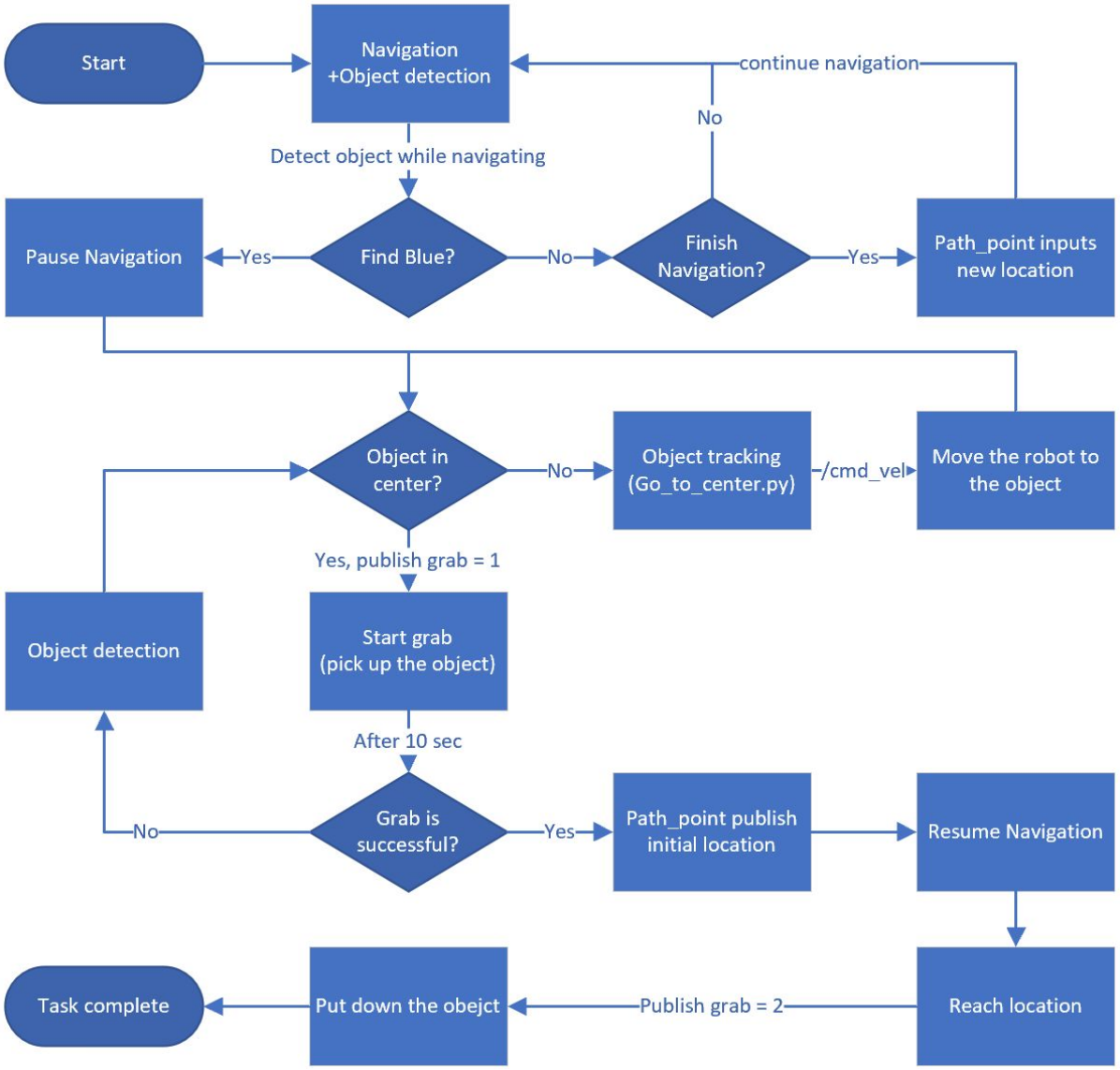


Node name: path_point			
Topic	Direction	Message type	Function
/clicked_point	Subscriber	geometry_msgs/Point Stamped	Record the point for searching
/path_point	Publisher	visualization_msgs/M arkerArray	Publish all the search points
/move_base_simple/goal	Publisher	geometry_msgs/Pose Stamped	Publish the navigation destinations
/move_base/result	Publisher & Subscriber	move_base_msgs/Mo veBaseActionResult	Indicate whether a navigation is finished
/find_blue	Subscriber	geometry_msgs/Point	Indicate of the blue target appeared in cam

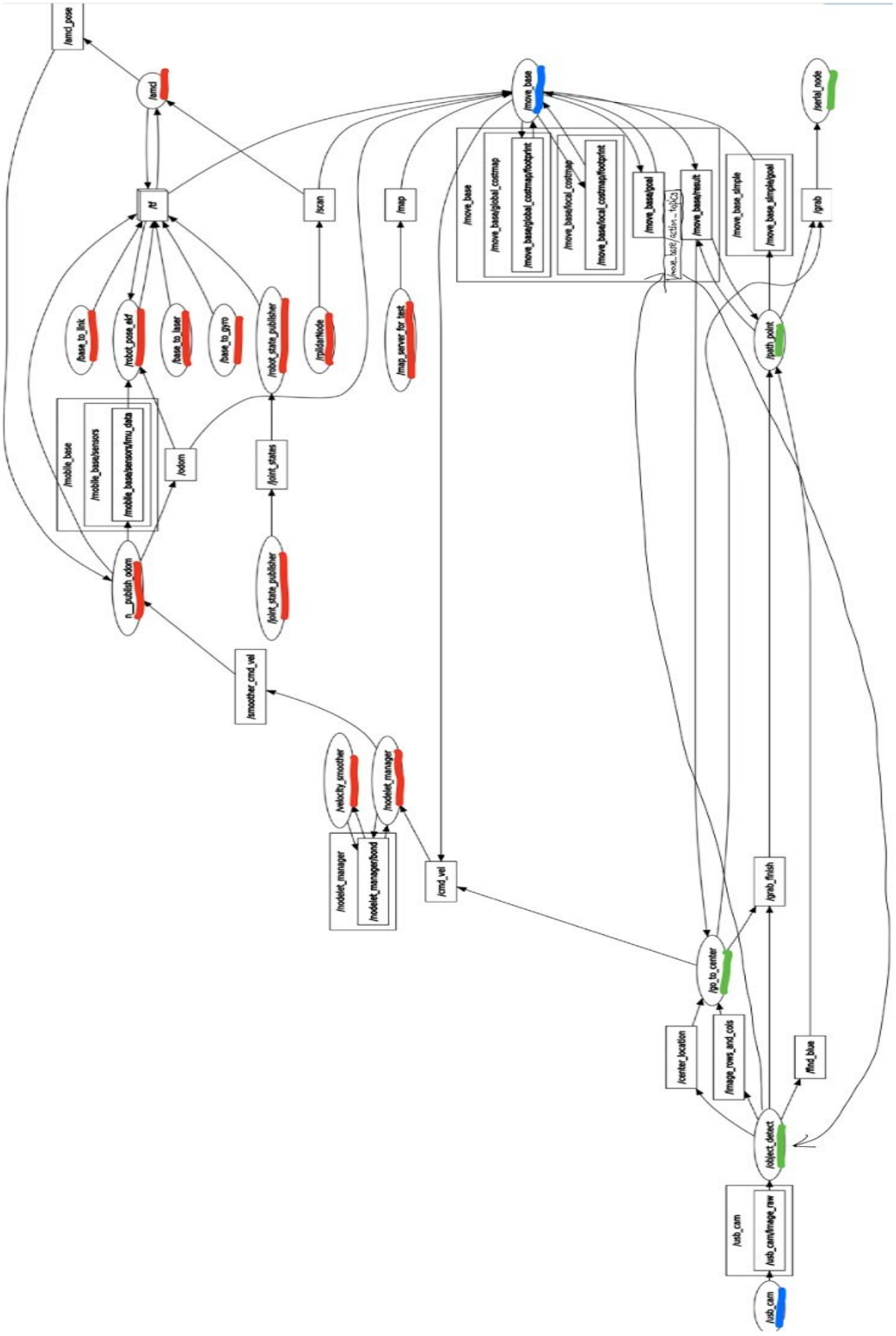
/grab_finish	Subscriber	geometry_msgs/Point	Indicate if the robot grab the object successfully
/grab	Publisher	geometry_msgs/Point	Ask the robot to put down the object in the end



The logic flow of this project:



The information flow of the project Rqt_graph:



Node Explanation:

Node	Provider	Description
/nodelete_mangaer	Huanyu	Subscribe to required velocity and use it to move the base
/move_base	ROS	1.Get goal input by node path_point and navigate the robot to the goal 2.Cancel navigation when receiving the request 3.Indicate if a point is reached or not by /move_base/result
/usb_cam	ROS	publish the topic contains the real-time image information
/obejct_detect	ourselves	This is the node created by vision.py. It has 2 functions: 1.Subscribe the image message in /usb_cam and publish the object center position to /center_location 2.indicate if the target is grabbed successfully by /grab_finsih
/go_to_center	ourselves	This is the node created by go_to_center.py. 1.subscribe to the topic (/center_location) and publish the relative velocity topic (/cmd_vel) to track the object 2.order the arm to pick up the object(/grab=1)
/serial_node	Arduino	Arduino board communicated with Jetson Nano through “rosserial” and showed in rqt as /serial_node. This node subscribes to the topic (/grab). If the node receives X=1 in /grab, the robot will perform the grab action. If the node receives X=2 in /grab, the robot will perform the put down action.
/path_point	ourselves	This is the node created by show_mark.py. 1.record and publish searching points for navigation to the topic (/move_base_simple/goal) 2.make the robot return to initial position (/move_base_simple/goal) when grab finish (/grab_finish=1).

Topic Explanation:

Topic (msg type)	Publisher	Subscriber	Description
/cmd_vel (Twist)	Move_base & go_to_center	/nodelet manager	Contains three linear velocity and three angular velocity
/find_blue (Point)	/object_detect	/path_point	If X=1, it means the blue object exists in the video frame. The only value X can have is 1. Only published one time in the whole rescue mission.
/grab(Point)	/go_to_center & /path_point	/serial_node	If X=1, it means the robot can perform the grab action. If X=2, it means the robot can put down the object.
/center_location (Point)	/object_detect	/go_to_center	It contains the object center position in the frame.
/grab_finish (Point)	/object_detection	/go_to_center & /path_point	If X = 1, it means the grab task is finished.
/raw_image (image)	/usb_cam	/object detect	Original image captured by cam
/opencv_image (image)	/object detect	N/A	Image proceed by openCV, used for testing and visualization on PC
/image_rows_and_cols (Point)	/object detect	/go_to_center	The height and width of image
/move_base/result (MoveBaseActionResult)	/move_base &/path_point & /object detect	/path_point	If status = 3, navigation of current point is complet. If status=1, robot is going to the next goal position
/move_base_simple/goal (PoseStamped)	/path_point	/move_base	Destination for navigation

Action:

Action	Client	Server	Service
SimpleActionClient<ActionSpec>::cancelGoal()	/object_detect	/move_base	Cancel current goal and stop navigation

Limitation & Further development

- Remote control range problem
- Servo fast reset problem
- Self-Mapping
- Use Better Servo to Judge Grab Success
- Improve Battery System
- Improve Vision
- Improve Integrity of Robot Structure
- Control Grabbing Process base on Vision
- Implement to Logistic
- A more integrated structure

Command

Step1: Generate a map

roscore

roslaunch huanyu_robot_start Huanyu_robot_start.launch

roslaunch huanyu_robot_start gmapping_slam.launch

roslaunch turtlebot_teleop keyboard_teleop.launc

rviz

cd robot_ws/src/hunayu_robot_start/map >> open terminal

rosrun map_server map_saver -f map_name

Step2: Rescue robot

cd robot_ws/src/huanyu_robot_start/launch

gedit navigation_slam.launch >> change map filename

roslaunch huanyu_robot_start Huanyu_robot_start.launch

roslaunch huanyu_robot_start navigation.launch

rviz

roslaunch usb_cam usb_cam-test.launch

rqt_image_view

rosrun obect_detect vision.py

rosrun object detect go_to_center.py

rosrun huanyu_robot_start show_mark.py

rosrun rosserial_python serial_node.py /dev/ttyUSB0

Packages

Package			
Number	Name of the package	Major Provider	Description
1	huanyu_robot _start	Huanyu	Robot driver for base control, it communicates with Hibot (STM32) We add show_mark.py in it for recording navigation goals
2	hunayubot_description	Hunayu	Detail dimension of hunayu robot and transformations between components >> used in mapping and navigation
3	turtlebot_teleop	ROS	Keyboard control of robot velocity >> used in mapping
4	rplidar_ros	ROS	rplidar driver: get lidar sensory input >> used in mapping and navigation
5	robot_pose_ekf	ROS	Estimate the 3D pose of a robot, based on (partial) pose measurements coming from different sources >> used for mapping and navigation
6	slam_gmapping	ROS	Mapping
7	Navigation melodic	ROS	Navigation
8	rosserial_python	ROS	Connect Arduino to ROS
9	usb_cam	ROS	Camera driver, capture real-time image >> used for object detection and tracking
10	Object detect	Ourselves	Object detection and tracking
11	publisher	Ourselves	(For testing only) publish message to topic /grab and /grab_finish
12	opencv_move	Ourselves	(For testing only) object detection and tracking in python, independent from ROS

Note

1. Inside package object_detect/src, we add a file called find_hsv.py, it is used for finding the lower and upper hsv boundary for the target (the boundary values are then used for color detection)