Navigation intérieure en temps réel pour les véhicules robots

- ZHANG Aihua
- PENG Qianbi

I. Vue globale système

Ce projet consiste à réaliser une voiture avec navigation automatique. Ce système est exploité dans l'environnement ROS sous Linux.

Après avoir connecté et piloté le lidar et la caméra dans le système ROS, nous avons divisé le projet en deux parties. La première étape consiste à localiser et à cartographier. La deuxième étape est la navigation autonome et l'évitement dynamique des obstacles.

- **La première étape : cartographier l'environnement, détecter les objet**

Nous utilisons l'algorithme SLAM pour le positionnement et la cartographie avec le logiciel RVIZ pour afficher le map.

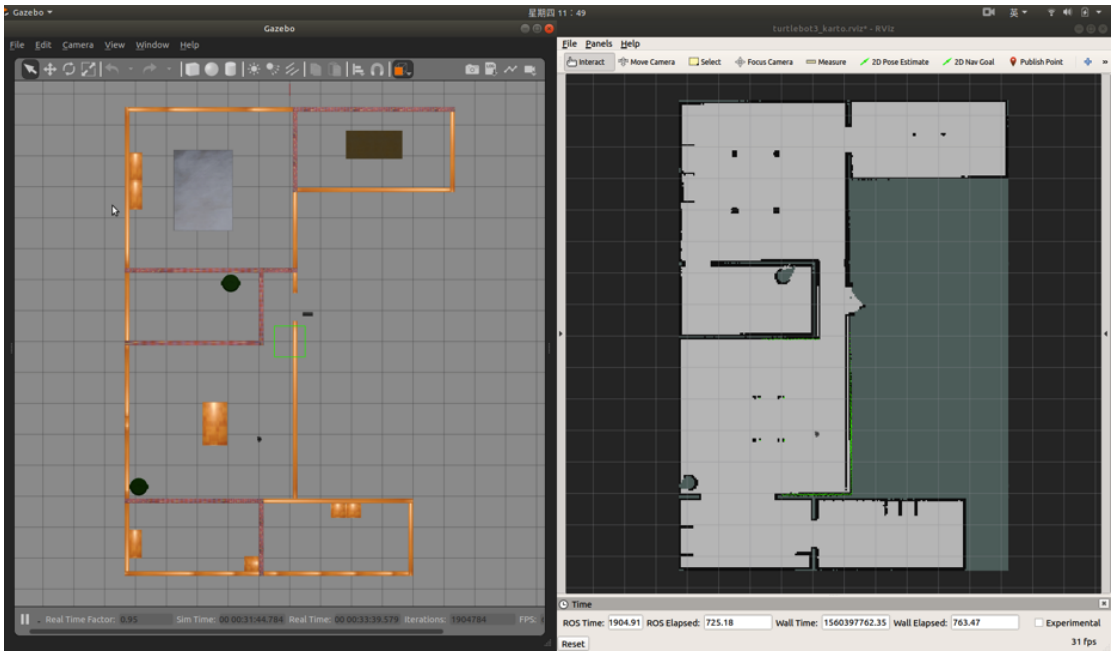
Le laser SLAM et le visuel SLAM sont les plus couramment utilisés actuellement. Dans ce projet, nous utilisons directement Google cartographer de laser SLAM pour gérer les conditions intérieures.

En plus des fonctions de base ci-dessus, nous avons également ajouté l'algorithme YOLO dans le traitement de la vidéo pour détecter les éléments dans la vidéo. Pour les photos ou vidéos existantes, nous utilisons directement l'algorithme yolo pour détecter les éléments. Pour la vidéo en temps réel, nous utilisons l'algorithme Darnet (YOLO sous ROS) pour détecter les éléments.

- **La deuxième étape : simulation de navigation**

Lorsqu'il n'y a pas de matériel de robot, nous utilisons Gazebo, une plate-forme de simulation physique de 3D pour simuler la navigation.

Nous pouvons créer un monde de robot gratuitement dans Gazebo, ce modèle peut simuler la fonction de mouvement du robot et les données de capteur du robot. Et ces données peuvent être affichées dans RVIZ. Le processus de mappage est comme ci-dessous:

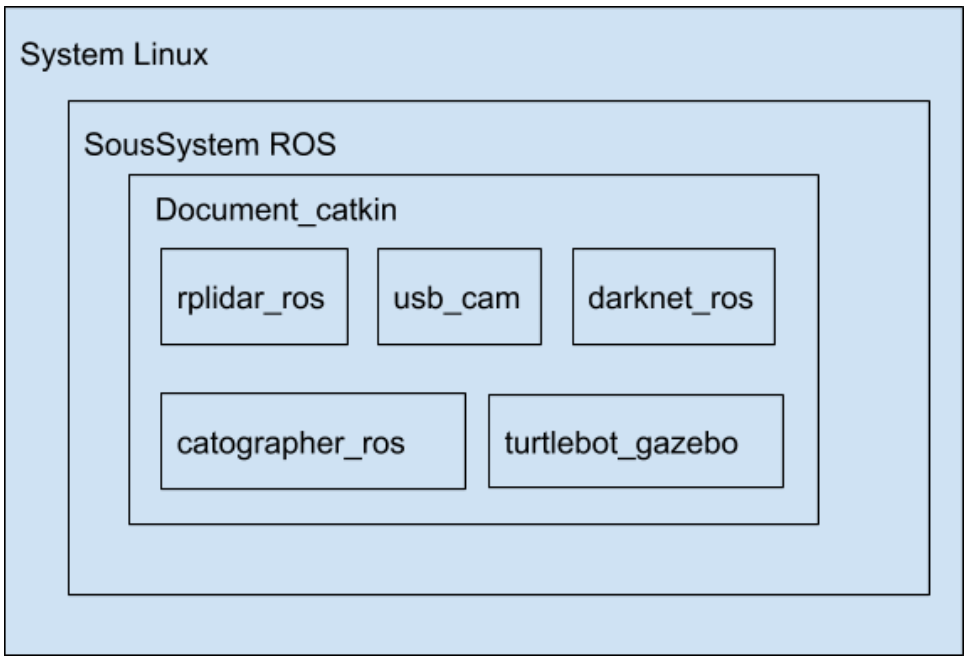


La gauche est le modèle de simulation de Gazebo, et la droite est l'effet de rendu montré dans RVIZ. Une fois la carte est créée, nous l'enregistre en tant que fichier .yaml. Nous pouvons ensuite l'utiliser pour effectuer une simulation de navigation autonome sur Turtlebot.

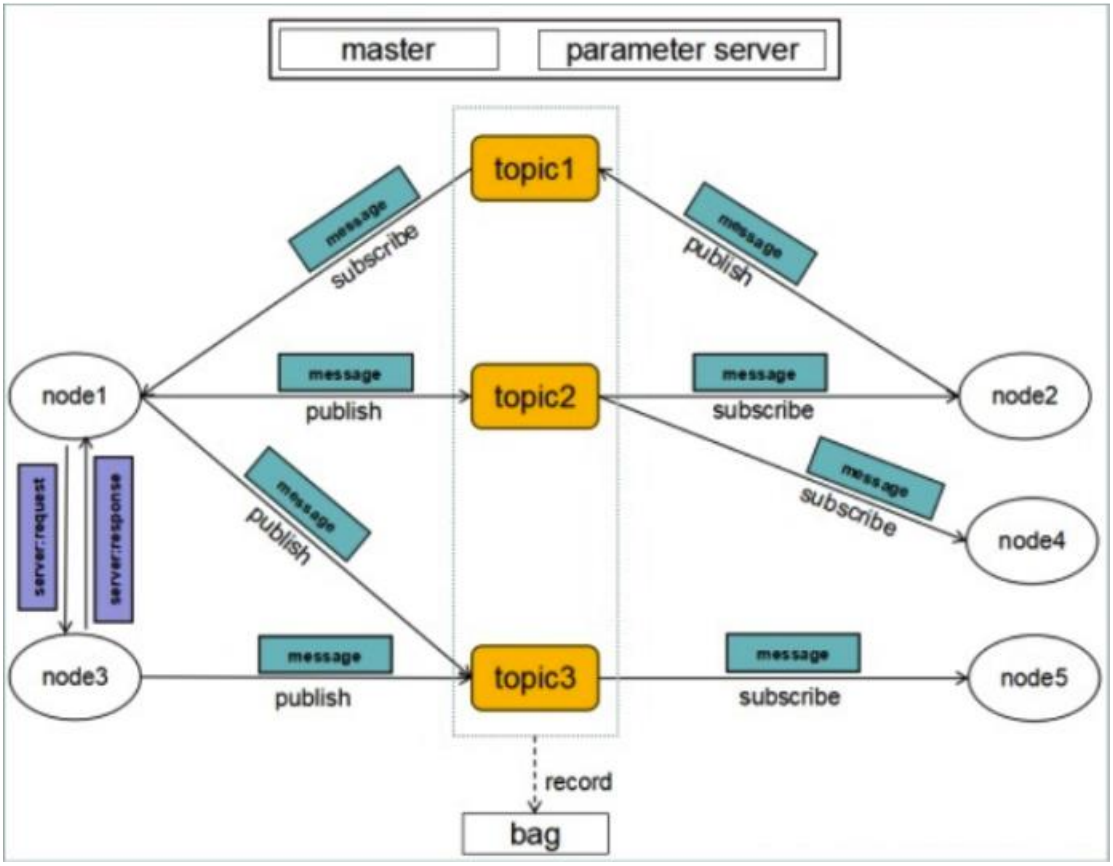
C'est-à-dire, définissez un emplacement cible sur la cartographie, et le turtlebot planifiera automatiquement le chemin, puis se déplacera lentement vers la position cible.

Après avoir mis en œuvre la première étape de ce projet, nous pouvons également enregistrer le graphique en tant que fichier .yaml, puis utiliser le gazebo pour naviguer.

Structure du système d'installation：



Après avoir installé le système Linux, le système ros, google cartographie, RP Lidar et le package de pilote de caméra sur la carte et l'ordinateur, Le PC distant peut utiliser rostopic pour obtenir les données du lidar et de la caméra via wifi. Le théorique sur rostopic est au-dessous:



Dans ce projet, notre carte exynos est le maître et l'ordinateur est l'esclave. Comme expliqué graphiquement dans l'image ci-dessus, Le système possède plusieurs nœuds qui réalisent différentes tâches. Les nœuds se communiquent par un mécanisme qui s'appelle topic. Un fichier bag collecte toutes les informations que l'on veut sur topics qui permet de les réutiliser ultérieurement.

II. Composants matériels

- **Carte exynos**

La carte exynos permet de lancer la distribution de linux (Ubuntu mate 16.04) qui sert à notre environnement de travail. Le ROS est installé sous linux pour faciliter le développement du robot. Nous définissons la carte comme le master, qui est responsable du démarrage des nœuds et de l'envoi des topics.

- **Lidar**

RPlidar est connecté au robot par le port USB en utilisant le package RPlidar conçu pour le système ROS.

- **Caméra**

Le package usb_cam sous ros nous permet de publier des topics sous différentes formes (raw, compressed, etc.). La vidéo compressé est plus facile à envoyer mais ne peut pas être traité par l'algorithme YOLO. La forme raw est adopté à la fin.

- **Module wifi**

L'ethernet et le wifi sont pas sous le même réseau à notre école et pour la portabilité, nous utilisons deux modules wifi qui réalisent la communication entre le PC et le robot.

- **Cart SD**

La carte SD contient la distribution de linux qui est notre environnement de travail.

- **PC**

Le PC représente le slave qui collecte toutes les information envoyées par la master. La vidéo est passée à l'algorithme pour distinguer les objets. Les nuages de points sont traitées par le google cartographer. Toutes ces informations traitées sont regroupées dans le logiciel graphique RVIZ.

III. Composants logiciels

- **Linux**

Linux est l'environnement de travail qui est installé dans la carte SD.

- **ROS**

ROS (Robot Operating System) est un système qui fournit différentes fonctionnalités : abstraction du matériel, contrôle des périphériques de bas niveau, mise en œuvre de fonctionnalités couramment utilisées, transmission de messages entre les processus et gestions des *packages* installés.
Les packages existants pour le lidar et caméra nous facilite le développement de notre robot.
Le système nous permet de communiquer plus facilement par topics entre plusieurs machines.

- **RIVZ**

RVIZ est un outil de visualisation en 3D qui nous permet de regrouper toutes les informations dans une interface graphique. En le configurant, nous pouvons directement afficher notre carte scannée et la vidéo traitée en temps réel suite du déplacement de notre robot.

- **Google gartographer**

Cartographer est un système qui fournit une localisation et une cartographie simultanées en temps réel (SLAM) en 2D et 3D sur plusieurs plateformes et configurations de capteurs.
Les algorithmes SLAM combinent les données de divers capteurs (par exemple LIDAR, IMU et caméras) pour calculer simultanément la position du capteur et une carte de l'environnement du capteur. Ça nous permet de créer la carte en exploitant le lidar à partir des nuages de points.

- **YOLO & Darknet_ros**

You only look once (YOLO) est un système de détection d'objets en temps réel.
Darknet_ros s'agit d'un package ROS développé pour la détection d'objets dans les images de la caméra en utilisant l'algorithme YOLO. Ça facilite l'intégration de YOLO dans notre système.

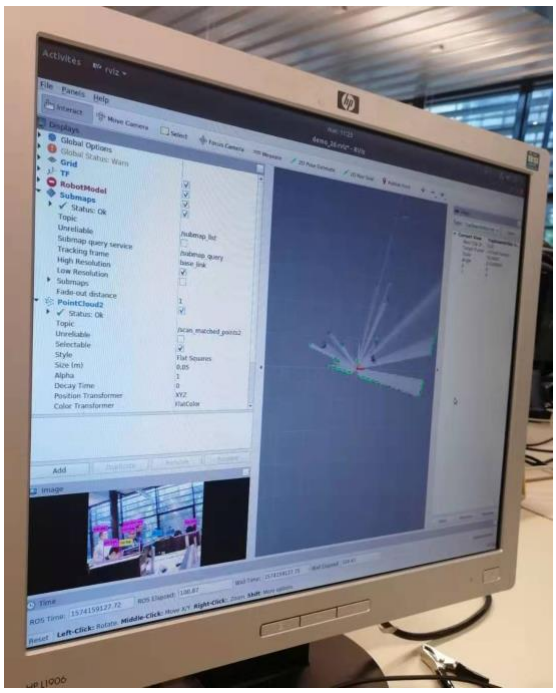
- **Tuetlebot_Gazebo**

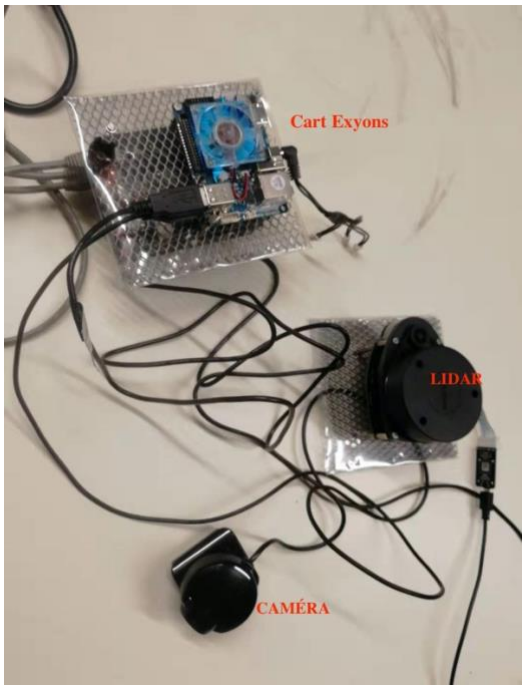
TurtleBot est un kit de robot personnel avec un logiciel open source. Le kit TurtleBot comprend une base mobile, un capteur de proximité 2D/3D, un ordinateur portable ou un ordinateur monocarte et le kit de matériel de montage TurtleBot.

Gazebo est un simulateur 3D, cinématique, dynamique et multi-robot permettant de simuler des robots articulés dans des environnements complexes, intérieurs ou extérieurs, réalistes et en trois dimensions. Globalement, Gazebo est idéal pour expérimenter comment les robots interagissent avec leur environnement.

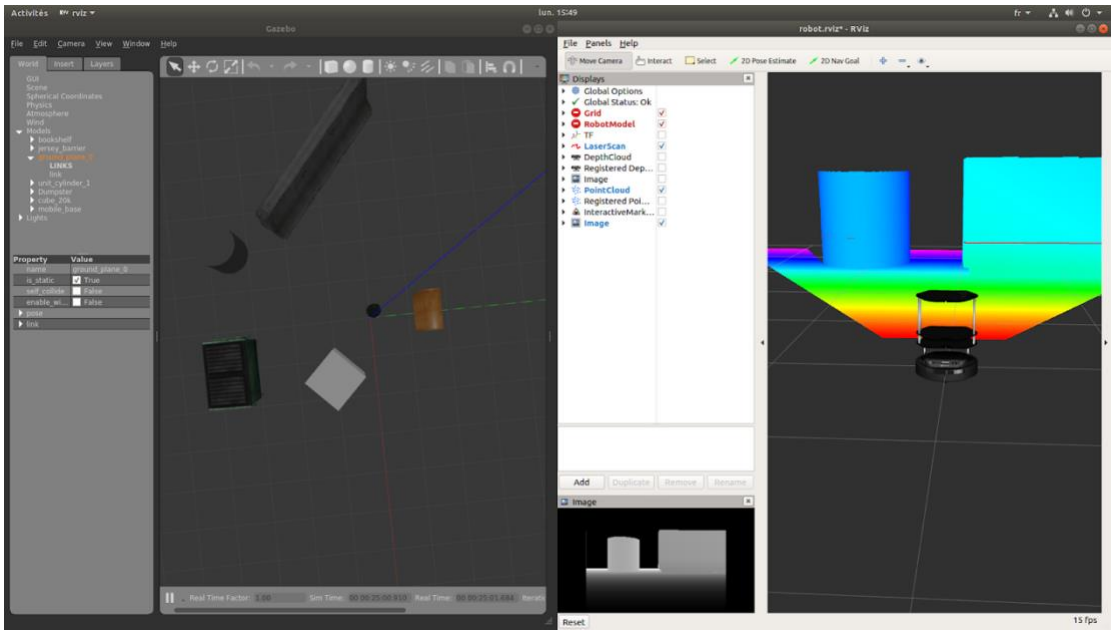
TurtleBot est un robot mobile dont l'objectif principal est de permettre à TurtleBot de détecter l'environnement sans heurter d'obstacles et de reconnaître la cible. Gazebo a créé l'environnement pour Turtlebot.

IV. Results





Après avoir bien intégré la caméra et le lidar dans le système, nous avons traité la vidéo reçue et les nuages de points respectivement par l'algorithme YOLO et Google Cartographer. Nous pouvons bien voir les cadres et les étiquettes autour de l'objet identifié. En paramétrant les paramètres de Google Cartographer, nous pouvons aussi suivre l'évolution de la carte scanné dans le logiciel RVIZ.



Nous pouvons créer un environnement de robot via un gazebo et afficher l'image dans RVIZ, mais pour notre propre fichier yaml, le robot ne peut pas se déplacer et ne peut pas naviguer automatiquement jusqu'au point cible.