ISTY UVSQ PARIS SCLAY

RobCup Rescue 2023 Team Description Paper TBRMS

Mendy Alphonse, Bruxelle Antoine, Aymeric Kwan, Tsakas Achille, Schroeyers Adrien

Information

Team Name:	TBRMS
Team Institution:	ISTY UVSQ
Team Country:	FRANCE
Team Leader:	Mendy Alphonse
Team URL:	

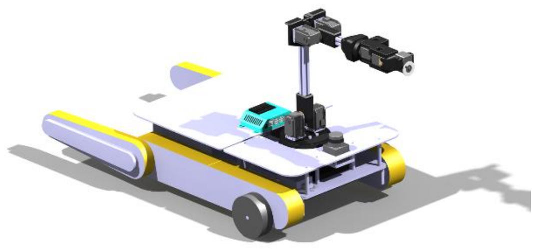

Figure 1 Rescue TBRMS

Abstract

This article details the configuration of the TBRMS robot planned to participate in the RoboCup Rescue 2023 challenge. The team's primary goal is to develop a robust semi-autonomous system that directly assists the operator in controlling the robot by returning the vision mapping data needed to explore rough or inaccessible terrain. . To overcome these challenges, we had to make choices related to sensors and high-quality electronic components. We tried to make the robot as user-friendly as possible to reduce the execution time compared to existing robot rescue systems.

I. Introduction

Our team is composed of five students from the Institute des Sciences et Techniques des Yvelines (University of Versailles Saint Quentin en Yvelines) and we are participating for the first time in the RoboCup Rescue League in order to evaluate our performance in a global challenge. Our main mission is to integrate sensors and electronic components on a mechanical platform provided by our engineering school.

We seek to leverage the main features available to the robot.
to push the limits of remote control of mobile robots. Our system is both semi-autonomous and hybrid due to human-computer collaboration. It is driven in differential mode and caterpillar mode depending on the type of terrain. Our platform is in line with search and rescue robots, it is distinguished by its speed of execution of tasks.

II. System Description

A. Hardware

Refer to table 2

1. Chassis

We use a mechanical platform with two steering modes: the Caterpillar mode is provided by belts that drive the chassis and two tracks. The base we use can support the weight of the arm and all the integrated components.

2. Arm and Manipulator

We designed and built a 6-axis robotic arm with a custom gripper. The degrees of freedom the manipulator provides, and the custom gripper will be able to complete the manipulation tasks.

The arm will be capable of precise motion reaching about 80 cm in every direction with an accurate end effector positioning.

We are working on developing our own gripper for the arm and we aim for it to be able to do all the tasks needed.

3. Computer

The robot uses a full desktop computer running Ubuntu 20.04 with ROS Noetic. We have an embedded computer (Jetson Nano) with Nvidia GPU TX1 and CUDA version 10.1 to improve performance in data computation from sensors.

4. Sensors

RGB-D (Intel D435) camera will be used to send real-time video feedback and point clouds to the operator. A Hokuyo UTM-30LX
LIDAR and a RGB-D camera will be used for mapping, localization and obstacle avoidance.

B. Software

The operator is controlling the robot with a Gaming Console controller such as Xbox or PlayStation controller or with the keyboard of any computer that as the robot application.
The Hokuyo and RGB-D camera will be connected directly to the embedded computer via USB.
Communication to the robot from the operator's station will be handled by a combination of UDP (User Datagram Protocol) and TCP (Transmission Control Protocol) calls over a single channel Wi-Fi connection. UDP will be used for teleoperation control and direct video feedback, while TCP will be used for less time sensitive data like map updates.

i. SLAM

Rescue robots are designed to navigate and operate in environments that are hazardous or inaccessible to humans. In order to navigate autonomously, these robots need to perceive their environment and create a map of their surroundings. One commonly used technique for achieving this is called Simultaneous Localization and Mapping (SLAM), which involves both building a map of the environment and estimating the robot's position within that map in real-time.

Hokuyo lidar sensors are often used in conjunction with rescue robots for SLAM. A lidar sensor emits laser beams in multiple directions and measures the time it takes for the beams to reflect off objects and return to the sensor. By measuring the distance and angle of the reflected beams, the lidar can create a 2D point cloud that represents the robot's surroundings. This point cloud can be used to create a map of the environment, and to locate the robot's position within that map using SLAM algorithms.

Hokuyo lidar sensors are a key component for SLAM on rescue robots, as they provide accurate and detailed information about the robot's surroundings, allowing it to navigate autonomously and effectively in hazardous or inaccessible environments.

ii. Path planning

The global map and point cloud data from the RGB-D camera will be used to plan the best path around obstacles. Obstacle avoidance will occur autonomously as the operator simply drives in the desired direction. When the flippers are added onto the drive train, the path planning will utilize them to navigate over obstacles, by setting their optimal angle for traversing over what is ahead of the robot. Development of this feature has already begun in simulation.

iii. Manipulator control

We are using the Moveit motion planner to perform path planning for the manipulator. Moveit allows us to pick a position in the attainable arm space and the arm will autonomously move and perform inverse kinematics. We can also control the end effector of the arm with Moveit.

The trajectory that Moveit generates will also avoid the obstacles in the manipulator space such as the sensors on top of the robot. We can also visualize the robot trajectory in a 3D virtual environment (fig2).

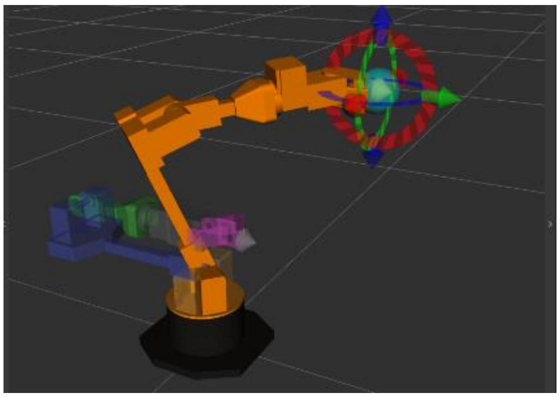
Figure 2 Moveit Interface

avoidance to prevent interference of the arm's joints with the real world is performed, which will be helpful for pick-and-place tasks. Additionally, we have extended MoveIt! to improve its path planning capabilities, and our system can move its end effector along arbitrary paths specified in 3D. This will be helpful for tasks in which precise and repeatable motion of the end effector is required, such as opening doors. The operator can select from a range of predefined poses and paths, and is also able to position the end effector in a 3D virtual environment (see Figure 2)

iv. Odometry for mapping and navigation

To navigate and map its environment, odometry is an input to the mapping and navigation algorithm. To do this, we have three different sensors: an Intel RGB-D camera (D435), an optical flow sensor, and an IMU. The odometry software is part of the ROS workspace embedded in a Jetson Nano. The RGB-D camera is used to calculate the robot's path using point clouds and 3D-3D & 3D-2D correspondences. The software is coded in CUDA to take advantage of the GPU's computational power of the embedded Jetson Nano. The visual odometry can suffer from drift, that's why it is combined with an IMU using an Extended Kalman Filter for loosely-coupled system. In addition, as input to the EKF, we have another data from the optical flow sensor that is pointed toward the ground, under the robot's chassis, to ensure path reconstitution reliability in dynamic environment.

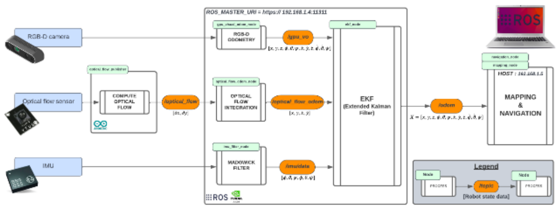

Figure 3 Odometry – Ros architecture

C. Communication

The communication between the robot and the operator station is done via a wireless network (Wi-Fi). The robot contains an internal Ethernet wired network used to interface with the Arduinos, the NVIDIA Jetson.

For the communication via Ethernet, we used custom UDP frames between ROS and the Arduinos for the communication between the various ROS machines we used ROSTCP. The UDP frames circulate at 10Hz. In addition, we use UART to communicate at a low level with other microcontrollers or sensors from the Arduinos. We have also included a radio frequency controller that serves as an emergency stop.

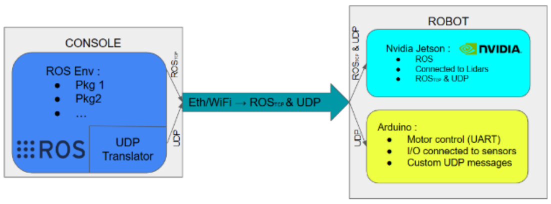

Figure 4 Communication Protocols

D. Human Robot Interface

The robot may be driven by one driver via the control interface. The interface is implemented as a Rqt node, with multiple plug-ins and provides a video feed from the on-board cameras, a click-and-drag map, an ROS message monitor, and a terminal. The GUI can be seen in Figure 5. Control will be semi-autonomous. The operator will control the direct the robot moves, but the robot will autonomously path plan for obstacle navigation. Manipulation will utilize inverse kinematics via MoveIt so that the operator can graphically set overall position goals relative to the robot in space that the arm will then perform (such as grasp an item, or place an item), rather than direct control of individual joints. On the outside of the robot chassis, a front panel is available to help control various parts of the robot. Ports for accessing the robot USB interface, display out, and charging the batteries allow for easier use of the robot without removing the arm to access the inside of the chassis.

III. Robot application

a) Setup

The robot application programmed in QT runs on a computer using Linux Ubuntu 20.04 and ROS Noetic

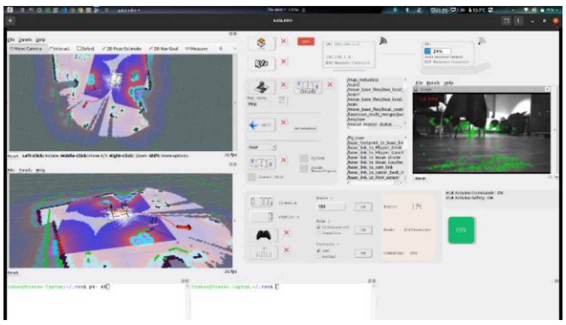

Figure 5 Set-up on Gui

With this robot application (image shown above) we can do several tasks.

Outputs:

- ❖ Selection between controller or keyboard control
- ❖ Flipper angle selection
- ❖ Caterpillar or differential mode selection
- ❖ Stopping or starting the robot
- ❖ Starting cartography, saving map
- ❖ Starting navigation with existing map (by selecting the desired map)
- ❖ Starting exploration (autonomous navigation and mapping)

Inputs:

- ❖ Control selected mode
- ❖ Flipper angle
- ❖ battery pourcentage
- ❖ Camera
- ❖ Navigation MAP (2D and 3D [point cloud]) and path/goal
- ❖ Connection status
- ❖ Robot feedback (custom debug error messages)

b) Mapping

Our robot is equipped with two lidars and cameras and can make a 2D mapping (using hector Slam) and 3D mapping (using OctoMap). (image XX :2D map made by our robot). We can make the mapping ourselves by controlling the robot manually, or with the exploration mode.

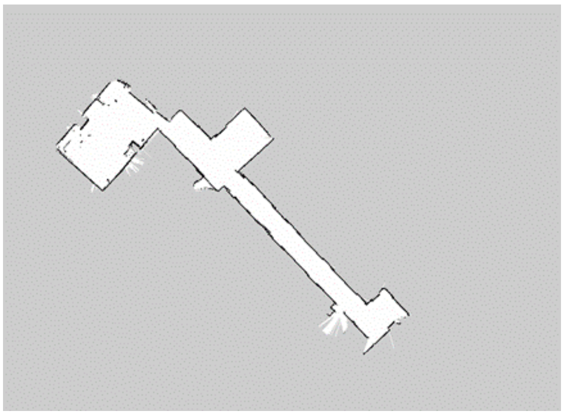

Figure 6 2D Map

c) Navigation mode

The robot is using the move_base_flex algorithm server controlled by a behavior Tree. We are using Teb_Local_Planner and DWA for local planning and A* for global planning. We have custom recovery behaviors for dynamic or not obstacles.

d) Exploration Mode

In this mode the robot is fully autonomous and does not need the computer application. When exploration mode is launched, the robot discovers his environment itself using the navigation mode and makes a 2D and 3D mapping; it can avoid dynamic obstacles and holes. Once the robot has finished the mapping it comes back to the starting point

IV. Conclusion

Finally, the team builds the software for testing. We are researching robots that are easy to operate and have high mobility on rough terrain. We would like to conduct these evaluations at RoboCup. When the results come out, I want to create a disaster site with a usable robot.

Tab 1 Softwares

Name	Version	License	Usage
Ubuntu	20.4	open	System
ROS	Noetic		System
OpenCV	4.2		Vision
Hector SLAM [3]	2020		2D SLAM
OctoMap			Probabilistic 3D Mapping
Image Transport			Video Streaming
Gazebo	11		Modeling
Rqt	5.12.8		GUI

Tab2 Hardware

Part	Brand & Model	Unit Price (euro)
Drive Motors	N/A	N/A
Drive gears	N/A	N/A
Drive encoder	N/A	N/A
Motor drivers	Sabertooth dual 25A motor driver	125*2
Batteries	Red Power LiPo 14.8V 6500 mAh	110*4
Computing Unit	Nvidia Jetson Nano	N/A
Wi-Fi Adapter	LinkSYS WRT54GL	N/A
LIDAR	Lidar Hokuyo URG-04-LX-UG01	1200*2
RGB-D Camera	Intel Realsense D425	378
Optical flow sensor	Thonflow 3901UY	20
IMU	BNO055	40
Cameras	N/A	N/A
Infrared Camera	N/A	N/A
Battery Chargers	Any LiPo battery charger	N/A
6-axis Robot Arm	Dynamixel MX-64 and AX-12 Servomotors	N/A
Drive Train	N/A	N/A
Operator Laptop	Any Laptop	N/A

References

[1] Review of visual odometry: types, approaches, challenges, and applications by Mohammad O. A. Aqel1*, Mohammad H. Marhaban2, M. Iqbal Saripan3 and Napsiah Bt. Ismail4

[2] CUDA C++ Programming Guide Release 12.1 NVIDIA

[3] VISUAL ODOMETRY for
MOVING RGB-D CAMERAS BY
Afonso Fontes
Universidade de Fortaleza
(UNIFOR)
José Everardo Bessa Maia
MACC
Universidade Estadual do Ceará.
(UECE)

[4] VINS-Mono: A Robust and Versatile Monocular
Visual-Inertial State Estimator
Tong Qin, Peiliang Li, and Shaojie Shen

Acknowledgement

The authors thank the staff of the engineering school Isty.
To provide support and space for our submissions
Possible. We would also like to thank Blazevic Pierre and Zaoui Mohamed for their technical assistance.

L'UNIVERSITÉ DE BORDEAUX PRÉSENTE
ROBOCUP23
BORDEAUX, NOUVELLE-AQUITAINE, FRANCE
LA PLUS GRANDE COMPÉTITION DE ROBOTIQUE ET D'INTELLIGENCE ARTIFICELLE DU MONDE
DU 4 AU 10 JUILLET 2023
PARC DES EXPOSITIONS DE BORDEAUX
2023.ROBOCUP.ORG