Control of the Omni-direction Robot and its Further Application

A project of the 2017 Robotics Course of the School of Information Science and Technology (SIST) of ShanghaiTech University

https://robotics.shanghaitech.edu.cn/teaching/robotics2017

Liu Junda liujd@shanghaitech.edu.cn,
Jiang Yiwen jiangyw@shanghaitech.edu.cn

1 Abstract

We design or reconstruct the omni-directional robot to do multiple tasks.

2 Introduction

Omni-directional mobile robots (OMRs) are typical holonomic robots. Compared to more common carlike or two wheels (nonholonomic) mobile robots, omni-directional mobile robots have the ability to translate and rotate simultaneously and independently. The maneuverability of this robot enables it to move as flexible as human beings in dynamic environmental applications.

3 State of the Art

3.1 Liu Junda

3.1.1 Development of an Omni-Directional Mobile Robot with 3 DOF Decoupling Drive Mechanism [1]

High mobility of robots are seential to enlarge their applications. In order to realize the cooperative motion of mobile robots, mechanisms for the high mobility is required. A new driving mechanism for holonomic omni-directional mobile robots is designed, which enabled 3DOF motion control by three correspondent actuators in a decoupled manner with no redundancy.

3.1.2 Improved Mecanum Wheel Design for Omni-directional Robots [2]

Conventional wheels are mechanically simple, have high load capacity and high tolerance to work surface irregularities. However due to their non-holonomic nature, they are not truly omni-directional. This paper proposes an improved design for a Mecanum wheel for omni-directional robots. This design will improve the efficiency mobile robots by reducing frictional forces and thereby improving performance.

3.1.3 Omni-Directional Mobile Robot Controller Design by Trajectory Linearization [3]

In this paper, modeling and nonlinear controller design for an omni-directional mobile robot are presented. Based on the robot dynamics model, a nonlinear controller is designed using the Trajectory Linearization Control (TLC) method. The TLC combines nonlinear dynamic inversion and linear time-varying eigenstructure assignment in a novel way, and has been successfully applied to missile and reusable launch vehicle flight control systems.

3.1.4 Omni-Directional Robot and Adaptive Control Method for Off-Road Running [4]

Many mobile robots are used in factories and hospitals. Most of them are required to have high-quality running mechanisms. Up to now, many types of omni-directional robots have been proposed, but they can only move on a flat floor because these OMRs have a wheeled running mechanism installed with free rollers or balls. They cannot be used in outdoor situations due to the poor ability for running over uneven ground.

This paper presents an off-road omni-directional mobile robot (OOMR) which can run on an uneven road and obstacles. The robot is constructed with for crawler-roller-motor units and can alse be called a "roller-crawler type of omni-directional mobile robot." Each crawler-roller-motor unit can be driven independently and the motion of the robot can be controlled by the speed of each motor. A position and velocity control system is also designed for the robot. The robot can be automatically controlled to run in an optional direction and to track an orbit. The adaptive control method for the OOMR is also shown. The efficiency of the mechanism and the control method has been verified by many practical running tests and computer simulations.

3.2 State of the art of Jiangyw

Literature Integrating Mecanum wheeled omni-directional mobile robots in ROS.
Control system design of a teleoperated omnidirectional mobile robot using ROS.
Output feedback tracking control of the omni-directional robot based on an orientation observer.
Dynamic Modeling and Control of Omni Directional Mobile Robots.

Literature 1(Dynamic.....) The author designed a three wheeled robot to realize"movement in any direction without reorientation of the cart." And they build kinematic model of the robot without considering friction first, and introduce friction model and Lagrange equation to get the whole model.They used a random mass center to design the model and it can be improved by the real robot designer as well as the reducing of control effort.

Literature 2(Output.....) In this paper, authors studied the output feedback tracking control problem of omni-directional robots.First, they construct an observer to estimate the value of unknown orientation of the robot through switching and event triggering strategies. Compared to present works, the velocity of the robot is only required persistently exciting, and hence can go through zero, meanwhile, the acceleration measurement of the robot is unneeded. Then, they designed an output feedback tracking controller, guaranteeing that tracking errors asymptotically converge to zero. Finally, numerical simulations verify all obtained theoretical results.

Literature 3(Control.....) The whole controlling system consisted three part in hardware:Driving system, Sensor network and Robotic Arms.The software structure is detailed by dwelling deeper into ROS node network.this work has provided several results in the field of robotics due to the development of a customized mobile robot for maintenance application.

Literature 4(Integrating) This paper describes the full design and integration of Mecanum wheeled omni-directional robot platform in ROS system, the robot employs special Mecanum wheel to fulfill omni-directional motion capabilities. It divided in Hardware part and Software(in ROS) part. The whole robot is a car with four Mecanum wheels attaching to a base which is a box with Arduino controller modules and a embedded computer inside.The embedded computer is equipped with a quad-core 2.5G CPU which can connect WiFi 802.11 b/g/n network.With the Atmega2560 as the core, the Arudino MEGA 2560 is a control board, which is connected to the embedded computer through the USB interface, thus realizing communication with PC (data transmission).For range sensing, the platform uses KS103 ultrasonic modules with measuring range from 1cm to 800cm and accuracy of 1mm. Arudino MEGA 2560 is connected to the motor encoder through D22-D29 connector port, which is used to get feedback value from encoders.In addition, the robot is equipped with Xbox Kinect V1 and HOKUYO UG-04X-UG01 sensor, which are respectively for machine vision and laser navigation. Both the devices are directly connected to the embedded PC through the USB interface.The software part is based on ROS which has a very simple control flow to publish encoder

message and control the servo.In conclusion,this paper discussed the methods and processes of manufacturing the Mecanum wheel omni-directional robot under ROS environment. Its mechanical design, circuit design, software design basing on ROS is studied in detail, and the experimental evaluation is produced.

4 System Description

The robot we've made so far, is a robot under the control of the pid controlling system which can support not only keyboard/joystick mode but also automatic program. We operated on the existed omni-directional robot in STAR Lab and we finished to coordinate the PID control part of the motor of the robot. The PID constant is coordinate to 20, 0.79999 and 0 with the QPPs of 250000 which cut the whole power of the motor into 250000 piece which match the scale of the encoder data on the control board. In the ros program, there is a control constant distance to determine the shut down condition of the motor control, in the previous version of the ros program, the distance is set to be so large that the robot has the problem of stopping: the motor keep controlling the speed after the end of control on the keyboard or the program. At first , we tried to send a empty message to stop the robot because the lock system of the motor would keep the motor in a speed of zero if we dont send control message to the roboclaw, but it failed. So we read the former program thoroughly, we found that in every line of speed controling, the constant of distance is set to 1000, which let the motor to keep controlling after the encoder return the data of 1000 cycle of the motor have run, which caused a infinite controlling when the speed is zero. So we change the constant to a less one, and the problem do not exist any more. And with the keyboard and joystick we can control the robot to move in a certain direction and speed, and can move exactly with the effect of inertance in a high speed with the help of pid control. The system consists of: four Maxon DC motors and encoders, two roboclaw motor drivers and a Raspberry Pi with ROS environment. The control program is run in the Raspberry Pi, and control information is sent through the serial port.

4.1 ROS package in detail

roboclaw_control.cpp: This is the ROS package to receive twist message from other motion control program, then convert it into the serial port information and send the information to roboclaw motor driver. It uses wiringPi and wiringSerial libraries to use the serial port on Raspberry Pi. This package should be run before other controlling program is run. It consists of following parts:

chatterCallback: This function takes twist message and calculate the speed for each of the four motors, then it calls **set_velocity_limitdistance** to send the speed to each motor.

set_velocity_limitdistance: This function takes the motor number and its speed and turn them into the serial port message that meets the roboclaw's requirement. Then this function sends the message through the serial port.

crc: Roboclaw uses a CRC(Cyclic Redundancy Check) to validate each packet it receives. This function calculates the CRC number.

4.2 Other codes in detail

keyop.py: This is the program that receives keyboard input and turns it into twist message as well as sends the message. The key function in this program is **readKeys**. It uses the library *curses* to get keyboard input, basicly, the arrow buttons and the page up/down button, to control the robot to move around and rotate. This program runs in 100Hz and has a time-out value, which means if you are not controlling the robot for some time, it will stop automatically. It also limits the maximum speed of the robot, to prevent misoprations that make the robot move to fast.

joystick_op.py: This program receives joystick input and turns it into twist message. The key function is **readJoystick**, which uses a library called *pygame* to get joystick information. The basic operations are done by the left stick of the joystick and the LB/RB button, controlling the robot to move around and rotate. It's worth noting that the magnitude you push the joystick will influence the speed of the robot.

draw_*.py: These are programs that control the robot to follow certain trajectory. These programs are all based on a function called *coordinate_to_twist*. This function takes a pair of X coordinate and a pair of Y coordinate, for example, $(x1, x2)$ and $(y1, y2)$, then calculates the twist information for the robot to move.

Thus, with this function, given a squence of X coordinates and a squence of Y coordinates, the robot can follow the trajectory that the coordinates representing.

login.sh and **send_ip.py**: Due to the particularity of ShanghaiTech's network, if we want to use ssh to control the robot, we need to login on a webpage and get the ip address of the Raspberry Pi. However, it is inconvinient to use a monitor and a keyboard connecting to the Pi every time, so we use these two programs to login automatically and send Raspberry Pi's ip adrress to a specific email address. We put them into the rc.local of the Raspberry Pi, so they can be executed automatically on the startup.

5 System Evaluation

The test we did is to run the robot in a certain speed, the speed control is ok, the robot can run in a constant speed and in a straight line. The rotation of the robot also perform well, the circle draw by the robot has an error about 10% compare to the theory area. so we think it works perfectly as a omni directional robot

6 Conclusion

The Omni-Directional Robot can meet the requirement of working in limited space and hard environment because of its ability to move and turn swiftly taking less space. We can combine it with many existed robot type to create much more new and interestin topic after.

We've carried out making the robot follow a certain trajectory, however, we are not good at graphics. So, in the future, a program that turns a given trajectory into X, Y squences can make the robot capable of following any given trajectory. Also, the current remote control method is ssh, which doesn't include a GUI and can only use terminal to control. Maybe some advanced control method that includes GUI and other functions can be found out.

References

[1] C. Ye, H. Ni, and S. Ma, "Development of an omni-directional wheel with differential structure," in *2012 IEEE International Conference on Mechatronics and Automation*, Aug 2012, pp. 1633–1638.

[2] O. Diegel, A. Badve, G. Bright, J. Potgieter, and S. Tlale, "Improved mecanum wheel design for omni-directional robots," in *Proc. 2002 Australasian Conference on Robotics and Automation, Auckland*, 2002, pp. 117–121.

[3] Y. Liu, X. Wu, J. J. Zhu, and J. Lew, "Omni-directional mobile robot controller design by trajectory linearization," in *American Control Conference, 2003. Proceedings of the 2003*, vol. 4. IEEE, 2003, pp. 3423–3428.

[4] P. Chen, S. Mitsutake, T. Isoda, and T. Shi, "Omni-directional robot and adaptive control method for off-road running," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 2, pp. 251–256, 2002.