ECE 445

SENIOR DESIGN LABORATORY

PROPOSAL

REMOTE ROBOT CAR CONTROL SYSTEM WITH RGBD CAMERA FOR 3D RECONSTRUCTION

Team Members:
Yuhao Ge, yuhaoge2
Hao Chen, haoc8
Junyan Li, junyanl3
Han Yang, hany6

Tuesday 4th April, 2023

Contents

1 Introduction

1.1 Problem and Solution Overview (Whole section revised)

1.1.1 Problem

The existing remote control systems utilized in applications such as robotics and drones encounter difficulties when operating in complex environments, resulting in inadequate control over the machine. Moreover, for remote control, operators usually lack a comprehensive view of their surroundings, thereby increasing the likelihood of accidents. Therefore, it is essential to design a remote control system that enables the user to effectively maneuver the car in intricate surroundings, while providing a comprehensive view of the environment, even when the vehicle is beyond the range of visual perception.

1.1.2 Solution

The use of 3D reconstruction technology is becoming increasingly prevalent across various industries, including the automotive sector. Our proposed solution involves the utilization of 3D reconstruction technology[3] to provide users with a better understanding of their surroundings while controlling a car. To achieve this, we plan to equip the robot car with an RGBD camera, which will capture the surrounding data and transmit it to a remote server (personal computer). The 3D reconstruction module will then run on the remote server in real-time[2], generating a 3D model of the car's surroundings.

With our product, users will be able to control the robot car remotely via a joystick while also viewing the car and its surroundings from a third-person perspective[1]. The perspective can be easily shifted to allow for a better understanding of the car's surroundings.

1.2 Visual Aid

Subfigure 1: Environment Scenario

Subfigure 2: User Interface with Joystick Controller

Figure 1: Pictorial Representation of our Solution

1.3 Background

Remote control systems are commonly used in various applications, including robotics, drones, and industrial machinery. These systems allow operators to control the movement of a machine from a safe distance, which can increase efficiency and reduce risks. However, in complex environments, it can be difficult for operators to navigate and control the machine effectively. This is especially true when the machine is small and hard to see, such as a robot car.

At the same time, 3D reconstruction technology is becoming increasingly popular in various industries, including the automotive industry. It can provide a more comprehensive view of the surrounding environment, making it easier for drivers to navigate and avoid obstacles. Applying this technology to a remote control car can be a great way to explore its potential and provide users with an innovative and useful experience.

1.4 High-Level Requirement List(Whole section revised)

1. The robot car can be remotely controlled using a joystick. The control system should allow real-time control with a delay of no more than 200 ms.
2. Four-wheel independent driving is necessary to achieve omnidirectional movement for the car, so it can move in complicated environment.
3. The WIFI bandwidth between the Raspberry Pi and the server should reach 10 MB/s to guarantee the enough image information be transmitted.
4. An appropriate down sampling method should be proposed to transfer the 640*480 image to 480*360 while containing as much informations. An efficient and reliable 3D reconstruction algorithm should also be proposed to produce a reconstructed 3D model based on the 640*480 image or even lower.
5. To ensure real-time performance, the 3D reconstruction algorithm must be efficient, with a calculation time of less than 0.2 seconds for each frame and a frame rate of 5 FPS.

2 Design

2.1 Block Diagram

We basically divide our solution into two parts, a **remote server** and a **robot car platform**. The overview and requirements of the subsystems of these two parts are as follows.

2.2 Subsystem Overview

2.2.1 Car Subsystem

Our car subsystem contains a motor and wheel controller, a base for the Raspberry Pi, and a rotatable base for placing the camera. The car subsystem is a movable robot car platform that can accept movement commands sent from the Raspberry Pi to perform actions such as steering and acceleration. We will design a rotatable camera base for this system, which will be used to help the sensor system capture information about the room from various angles. In order to make the sensor system work well and consistently, we need to design our robot car platform carefully, especially the rotatable base.

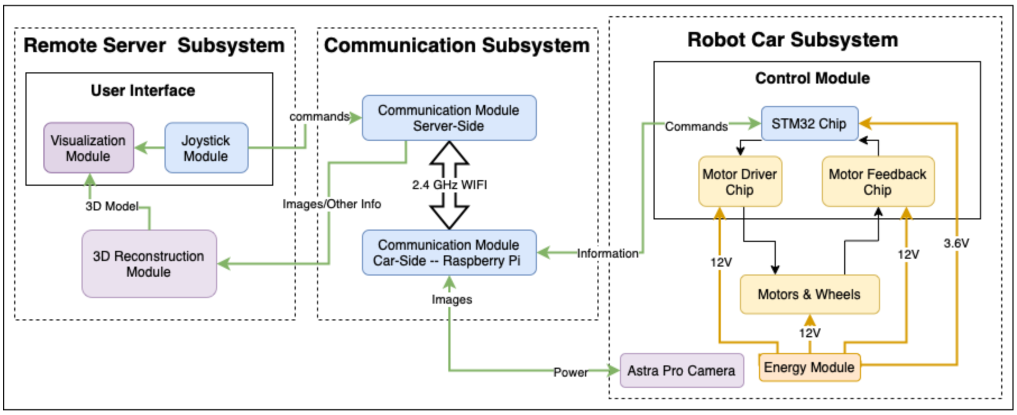

Figure 2: Block Diagram of our Solution

2.2.2 Sensing Subsystem

The main component of the sensing system is an RGBD camera, which provides colour images and corresponding depth information. In our design, the camera will be placed on a rotatable base so that it can capture 360 degrees of information without the car moving. The images captured by the camera will be sent to a remote server for further processing via Raspberry Pi and communication module. Information about the motion of the car can also be collected if needed.

2.2.3 Communication and Control Subsystem

We need a communication and control (C&C) subsystem to realize remote control and sensor signal transmission. We will use Raspberry Pi as the core of the C&C subsystem to provide communication for the remote server with the car subsystem and sensor subsystem. For example, RGBD signals from the sensor subsystem will be sent by Raspberry Pi to the remote server via Wi-Fi; the car motion commands sent from the remote server will be delivered to the corresponding motor by the control module.

2.2.4 Remote Server

The remote server is an important part of our project, as it will perform 3D reconstruction and provide an interface subsystem for remote control. We have subdivided the remote server into the following parts. First is the 3D reconstruction module, which uses OpenCV to process RGBD camera images and perform 3D reconstruction using algorithms such as SLAM (simultaneous localization and mapping) or structure from motion. Then there is the user interface subsystem, which can be developed using web technologies such as HTML, CSS and JavaScript. The user interface allows the user to control the robotic car remotely via a joystick and view a reconstructed 3D model of the surrounding environment in real-time. There is also a communication module for data transmission with the robotic car platform.

2.3 Subsystem Requirements

2.3.1 Car Subsystem

The car handles all the movement of the model and it provides a platform to set up the Raspberry and sensing system. The car should ensure that it can move stably around the room even with a sharp turn. In this case, we use four wheels with at least 5cm radius and the energy can supply 5(+/-) 0.1 V voltage lasting 30mins to motors. To make the car move more efficiently, the speed of the car should reach 5km/h and the motor activates the two back wheels. The motor is controlled by the Rasberry Pi which provides the power signal to the energy module to change the speed of the motor.

2.3.2 Sensing Subsystem

The sensing subsystem consists of an RGBD camera located on the platform of the car. It has a self-support energy system and it sends RGBD signal to the Raspberry Pi through a USB line. The rotating device is a cloud platform to enable the spinning of the camera.

2.3.3 Communication and Control Subsystem

Raspberry Pi is the brain of our car, so it must have the capacity to handle raw data from the RGBD camera in a real-time. It should have the ability to send the data to the remote server at a high enough frequency, such as 25 FPS. Also, it should receive the control signal at a high frequency so that we can control the car smoothly and it won't crash into a wall.

In order to control the car's direction and speed, the raspberry pi should output PWM signals to change the motor's speed preciously. GPIO in raspberry pi will be used to fulfill this purpose. We will use Ubuntu as the pi's operating system since we will also need to read data from the RGBD camera through USB. Then, after reading the data, we will send the data through WiFi using TCP protocol to the remote server. It will also receive the control signal through WiFi or Bluetooth and act as expected.

2.3.4 Remote Server

Remote Server is a powerful computer that can receive the received RGBD data, process it and perform the 3D reconstruction. The reconstruction result should be clear, and we should be able to separate two neighboring objects from each other, such as two chairs. We will also design a UI to show the reconstruction result and users can scroll and change views as they wish. The whole process will be smooth, which means that the reconstruction speed should be fast (at least 5 FPS on average).

2.4 Tolerance Analysis (Whole section revised)

There are two main topics related to the remote robot car control system. The first topic is the time delay, which can cause deviations in the car's motion, positioning problem, and even collision problem. The second topic is the RGBD image transmission, which is important to enable the reconstruction system to receive real-time RGBD image input.

2.4.1 Time Delay

The time delays in processing and communication of robot car control can cause deviations in robot car's motion, which can cause positioning problem and even collision problem.

The source of the time delay comes from the remote control pipeline. The joystick module processes and analyzes the joystick signal to generate motion commands. The commands are encoded and transmitted by the communication module via Wi-Fi with TCP protocol. The control module calculates the PWM of each motor based on the decoded motion commands. Although this pipeline is complex and it is difficult and tedious to analyze the delay in detail, we can give a maximum allowable delay range because the remote control itself requires low delay and high accuracy. We expect the remote control of the car has a maximum delay $\Delta_{max} t = 100ms$, otherwise the remote control system can crash due to the high latency.

Under the assumption of worst-case latency, and given the designed maximum speed of the car is $0.5 - 1.0$ m/s, the position drift of the car is limited to $10 - 20$ cm.

$$10 \text{ cm} < \Delta_{\text{max}} x = 2 \times v_{\text{max}} \times \Delta_{\text{max}} t < 20 \text{ cm} \tag{1}$$

The ideal latency $\Delta_{ideal} t$ for a good control experience is within 60 ms, in this case, the drift is improved to 6-12 cm. According to the RGBD camera operation guide, the appropriate distance between the depth camera and the objects is in the range of 0.8-8.0 m. Consider two working cases, the first is that the distance between the car and the object is within the working range of the depth camera, and the second is that the distance between the car and the object is less than the minimum working distance.

- In the first case, the depth camera works properly, so the car's surroundings are being modeled in real-time. The proof is as follows: our camera takes RGBD images at 30 fps, and assuming that the car moves at a maximum speed of 1m/s relative to the obstacle, the maximum displacement of the car between two successive images is 3.3cm. 3.3cm is much less than 0.8m and can be neglected. Therefore, we can proof that the surrounding environment can be reconstructed at a minimum working distance of 0.8m. Then, the 3D reconstructed model can in turn assist in positioning and correct the drift.
- In the second case, since the depth camera will likely fail to work properly, we must consider early warnings on the visualization page, such as prompting for obstacle avoidance. Besides, collision prevention like forcing a speed reduction for dangerous situations is also necessary in a real-world application. When the cart is monitored to be less than 1m from the nearest obstacle, the maximum speed should be limited to be less than 20 cm/s. Meanwhile, the RGB image might itself provide better visual information about the surroundings, especially for close obstacles. So, switching the view to RGB images may also be a desirable measure.

Based on the analysis of the positioning error caused by the delay, we can flexibly improve our system design. The first is to reduce the latency. Communication latency is the main source of latency, if the direct use of public network cannot meet the requirements of low latency, we can choose to use LAN hotspot, or use 5G communication to reduce the latency.

To avoid possible collision hazards caused by delay, we can warn on the visualization page when it is close to the obstacle by 1m, and force to reduce the speed of movement when the distance is less than 0.8m. We can even take over the control when the distance is too close, such as less than 20cm, to avoid a possible collision, if necessary.

2.4.2 RGBD Image Transmission

To enable the reconstruction system to receive real-time RGBD image input, it is important that we can control the RGBD image transmission rate to a suitable range, optimally between 10-24 FPS. The transmission rate depends on image size and bandwidth of the WiFi connection.

According to our preliminary measurement, the bandwidth for our school WiFi eduroam between Raspberry Pi and the remote server is roughly 7MB/s. The image output of the camera is maximum 480x640, and each pixel in RGB image takes 3 Bytes, while each pixel in depth image takes 1 Byte. As a result, it takes

$$480 \times 640 \text{ pixels } \times 4 \text{ Bytes } \approx 1.17 \text{ MB} \tag{2}$$

to store one RGBD image. If we adopt JPEG compression with highest quality, the image size can be reduced to half of the original size, which means that now one RGBD image will be 0.6MB. If we transmit it using our school WiFi, the estimated transmission rate is 11.6 images/s.

Notice that the estimation here is based on the assumption that we use the highest quality compression setting and use the school WiFi which we know that it is not very fast. There is a lot of room for us to improve the transmission rate, such as decreasing the compression quality, or making the Raspberry Pi itself to be the wireless access point so that we can have stronger and more stable WiFi signal and faster WiFi speed. Also, we can also downsample the image resolution by a factor of 2 and it will easily boost the transmission rate to 46.4 images/s. There is a trade-off between the tramsmission rate and the image quality, which will in turn affect the reconsutrction quality. We will find a optimal trade-off for our system and use case during our system development.

3 Ethics and Safety

3.1 Ethics

The ethical concerns for our RGBD reconstruction system include privacy and information security. The use of the RGBD camera for 3D reconstruction raises concerns about the privacy of individuals, as the camera can capture images and other data that can be used to identify individuals. We need to ensure that the people captured will be anonymized properly.

Additionally, information security concerns arise with the storage and transmission of data collected by the camera. If the data transmission and storage is insecure, hackers can get these sensitive data and sell them to evil institutions, bringing us and probably other people huge trouble.

The IEEE and ACM Code of Ethics both emphasize the importance of respecting the privacy of individuals and protecting their personal data. According to the IEEE Code of Ethics, engineers must "protect the privacy of others", and they must only use the information for legitimate purposes. Similarly, the ACM Code of Ethics, Section 1.6 Respect privacy, requires that computing professionals respect the privacy of others and protect the confidentiality of any data they access.

To avoid ethical breaches, we will implement appropriate security measures to ensure that the data collected by the RGBD camera is stored and transmitted securely, including using an encrypted data transmission protocol, deleting any intermediate and temporary data when applicable, and storing the reconstruction result under an encrypted folder. We will also seek informed consent from individuals whose data is being collected and ensure that their privacy is protected throughout the project.

3.2 Safety

Safety is the highest priority in our project, and we have recognized several potential safety issues, including the risk of hurting other people and damaging things if our car is out of control. If the car is not adequately controlled, it may lead to accidents or collisions. Also, the battery or the power system is also a very important part since it may cause short-circuit if the battery is not functioning properly and even an explosion could happen.

To mitigate safety concerns, we will follow government regulations, industry standards, and campus policies related to the use of remote control cars and unmanned vehicles. We will also ensure that appropriate safety measures are implemented, such as limiting the car's speed and operating it only in controlled environments. The power system will be checked frequently to ensure that it is in a normal state and it can function properly before we actually mount it onto our car. We will also implement fail-safes to prevent any potential accidents and collisions.

References

[1] M Aharchi and M Ait Kbir. A review on 3d reconstruction techniques from 2d images. In *Innovations in Smart Cities Applications Edition 3: The Proceedings of the 4th International Conference on Smart City Applications 4*, pages 510–522. Springer, 2020.

[2] Chen Liu, Jiaye Wu, and Yasutaka Furukawa. Floornet: A unified framework for floorplan reconstruction from 3d scans. In *Proceedings of the European conference on computer vision (ECCV)*, pages 201–217, 2018.

[3] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019.