The User Manual

Introduction

We strongly wish our product will help you. The main requirements from our clients are detecting the fire and transmitting the images/videos to the ground station wirelessly. Moreover, these two parts should be real-time.

We are pleased that you have chosen Wildfire Drone for your business needs. There is a strong need for the fire detection algorithm and wireless communication model, as evidenced by the fire detection and images/videos transmission. We provide for you here a powerful system for fire identification model via deep learning algorithm and the QPSK wireless communication system has been custom-designed to meet your needs. Some of the key highlights include:

- 93% fire identification accuracy for real-time image analysis
- Fire identification model has a powerful ability to keep the accuracy even there is a strong noise
- Fire segmentation algorithm could circle the fire correctly based on the thermal images even the environment temperature is close to the fire
- The wireless communication system we designed is not just a simple transmitter & receiver, it also assembled with the error correction and the noise reduction parts. After multiple tests and modifications, finally it can transmit our images data in an accurate mode.

The purpose of this user manual is to help you, the client, successfully use and maintain the Wildfire Drone product in your actual business context going forward. Our aim is to make sure that you are able to benefit from our product for many years to come!

Based on the statistics result from the National Interagency Fire Center (NIFC), wildfires exhaust 10 million acres of land in 2016 and brought $6 billion irrecoverable damages from 1995 to 2014 in the United States. Wildfires not only impact the wildlife, but more importantly endanger human lives. Therefore, early detection of wildfires before they get out of control is an urgent requirement. Wildfires are often initiated in remote forest areas where the common fire detection methods such as lookout tower stations fail to detect such fires in a timely manner. Moreover, conventional detection approaches can barely provide sufficient fire information about the precise fire locations, the orientation of fire expansion, etc. To detect forest fires, there are two general approaches using satellite images, and sensor networks. However, the satellites cannot provide real-time video or images since the quality of their images is highly impacted by weather conditions. Fire detection using wireless sensor networks is costly and high maintenance to cover wide forest areas. Manned aircraft can precisely survey a wide area in a short amount of time, however, this solution is costly and will endanger the life of pilots due to the high-temperature airflow and thick smoke.

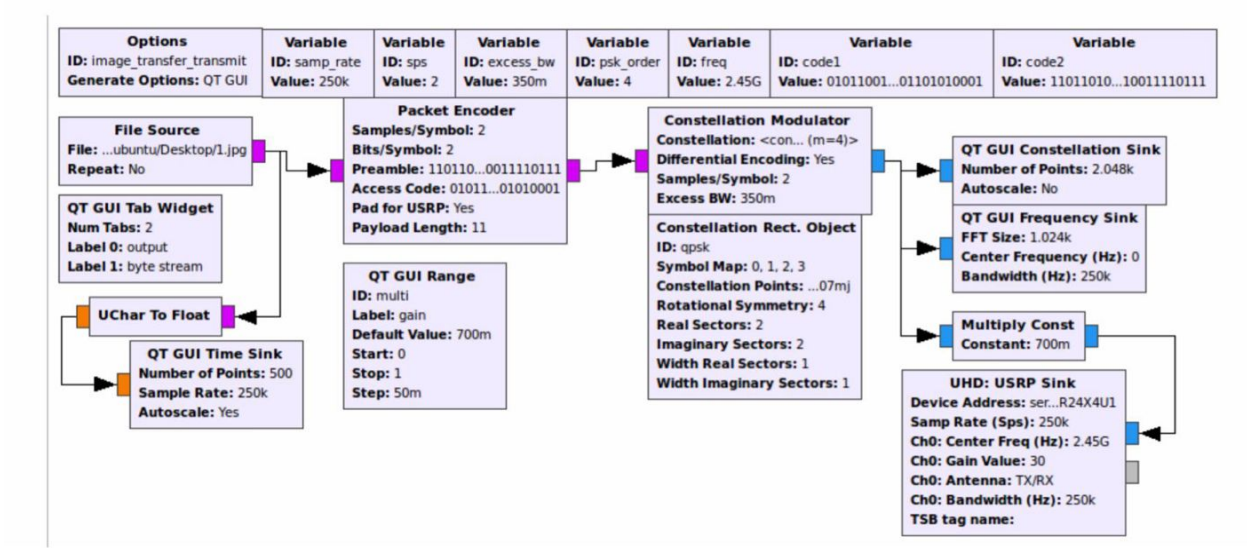
Unmanned Aerial Vehicles (UAV) have been recently utilized in wildfire detection and management as a low-cost and agile solution to collect data/imagery considering their unique features such as 3-dimensional movements, easy to fly, maneuverability and flexibility. The UAV networks can offer several features in such operations including tracking the fire front line, fast mapping of wide areas and damage assessment, real-time video streaming, and search-and-rescue. After this, the wireless communication could give us a chance to double check the images/videos so that the ground station could give the fastest and accurate information to the fire fighter.

Subsystem 1 - wireless communication

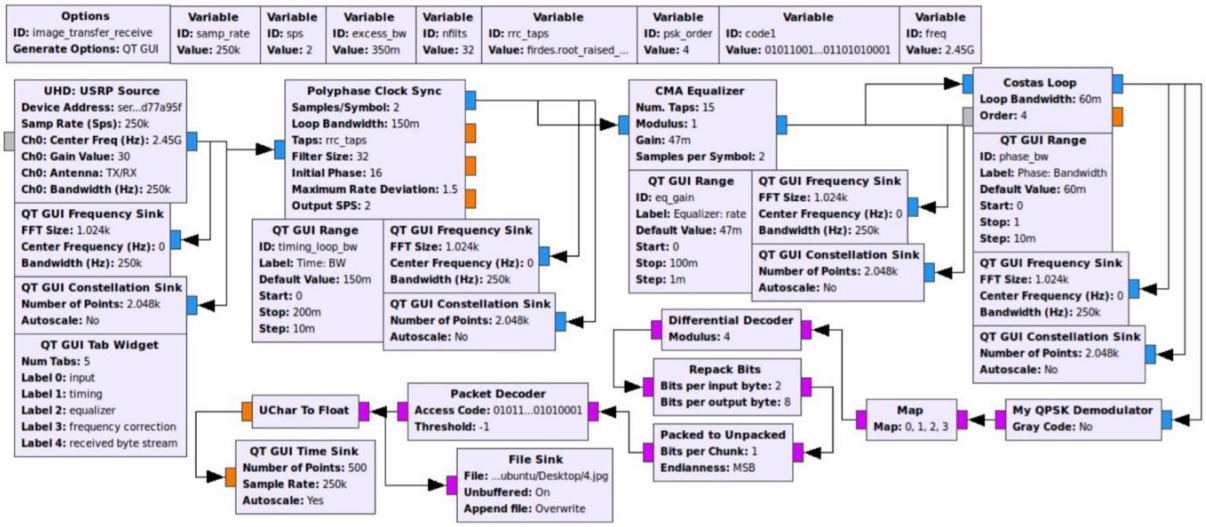
Overview

For the subunit who is in charge of the wireless communication part of the project, during our previous work, we have already designed and built the corresponding function blocks on the transmitter side and the receiver side using the software GNU RADIO, which means we have finished the basic goal of our project. However, due to the COVID-19, according to the suggestion of professor Afghah and our client, we need to stay at our dormitory to finish the remaining part, which includes the simulation of the whole transmission process. Unfortunately, the computers of our members are installed with the Windows system, but the software needs the Linux environment. In that case, we are now trying to install the virtual machine with the Linux system and test our function blocks on our own computer. In my view, the challenges in future may be the simulation of transmitting and receiving through a simulated channel. To realize the simulation, we should build a simulated wireless channel with practical characteristics, which means we need to consider the complex environment in practice where we process our wireless communication functions. To ensure a better performance of the communication system, we may also make changes to our two basic function blocks to make it more suitable to the new environment, or to gain a higher efficiency when they work.

The transmitter side:



The receiver side:



Subsystem 2 - image processing

Fire identification (Haiyu Wu)

One of the main parts of this project is to identify if there is a fire in the image/video captured by the camera. In this part, my approach is using a neural network. Neural networks are the most powerful tool in the image classification which is also fit in our case.

However, normally, convolutional neural networks (CNN) needs to be supported by a high computational equipment which is impossible for us to use in an embedded device. Fortunately, there is a lightweight network structure named mobilenet. It's structure allowed us to embed it into a Raspberry Pi 4 and still have a high classification accuracy.

When we tried to train the neural network, there was another challenge. To make our neural network have high classification ability, we need to use a huge image dataset (like over 100 thousand images). However, our project is related to wildfire which is unique and special so that it is hard to collect such a huge dataset. Therefore, we tried to use another useful tool called transfer learning. It is pre-trained by using the IMAGENET data source, then we use our special dataset to re-train it to improve the classification ability in our own case.

Network structure

Input	operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	256	-	
$1 \times 1 \times 256$	conv2d 1x1	-	2	-	

Table 1.

Normal image:

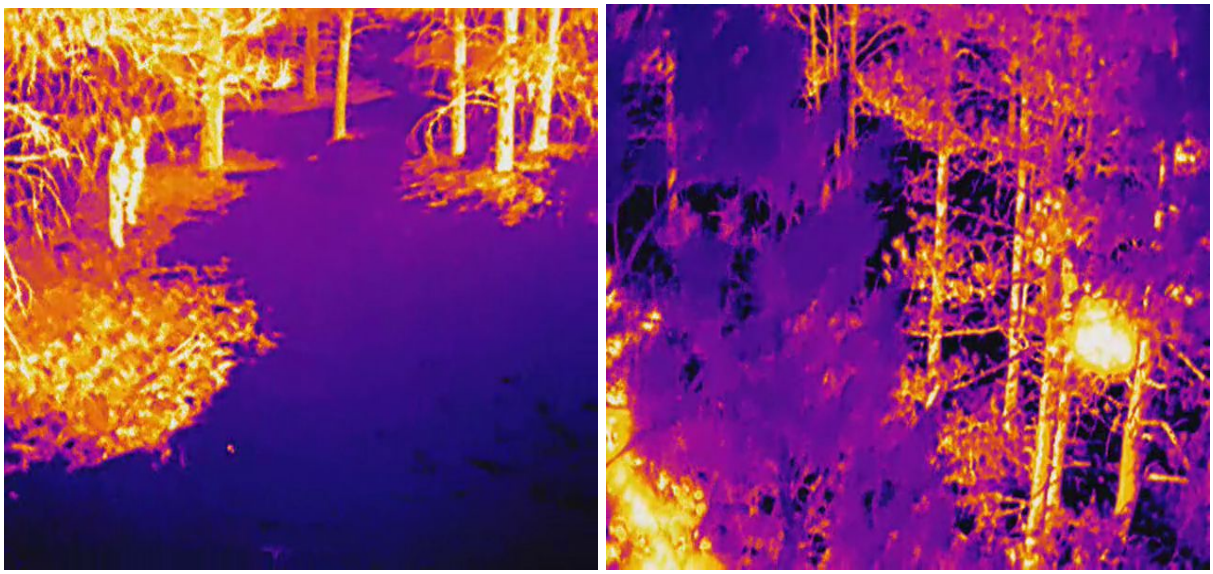
- Number of images: 1102 with fire and 1102 without fire for training. 305 with fire and 49 without fire for testing.
- Training time: 30 mins with GTX 1080ti.
- Identification time: 0.025s per batch (16 images)
- Resolution: 224 * 224
- Accuracy: 96.4%

Thermal image:

- Number of images: 998 with fire and 969 without fire for training. 224 with fire and 112 without fire for testing.
- Training time: 15 mins with GTX 1080ti.
- Identification time: 0.032s per batch (16 images)
- Resolution: 224 * 224
- Accuracy: 64.2%

Results:

1. For the normal images, our network works well in fire identification and the speed is very fast which could be real-time.
2. For the thermal images, the accuracy is really low. The main reason we figured it out is when we take the thermal videos with a FLIR camera, it will automatically adjust the color (shown below). The features of these two images are really close, which is hard for our computer to extract the unique feature for fire and no fire.



Without fire Fire

Image 1.

<u>Feature based image segmentation (Oshan)</u>

The second part of the image processing aspect of our project was image segmentation. This would pinpoint the location of the fire to the user on the image that is presented to him/her. We decided to use Matlab due to it's built-in image processing library. We decided to use Matlab because we primarily wanted to implement the image segmentation on the ground station, However our Matlab code can easily be adapted into Octave which will allow us to implement it in the Raspbian OS.

I exclusively decided to use thermal images with the "Fusion" color palette (Figure 1) for image segmentation, as the diversity of colors helped identify hotspots more effectively . I decided I had two different approaches first by trying to use the complete RGB and the second approach was to convert the original image to a binary image.

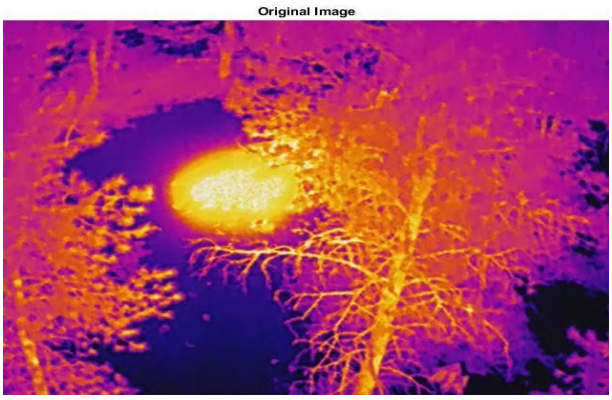
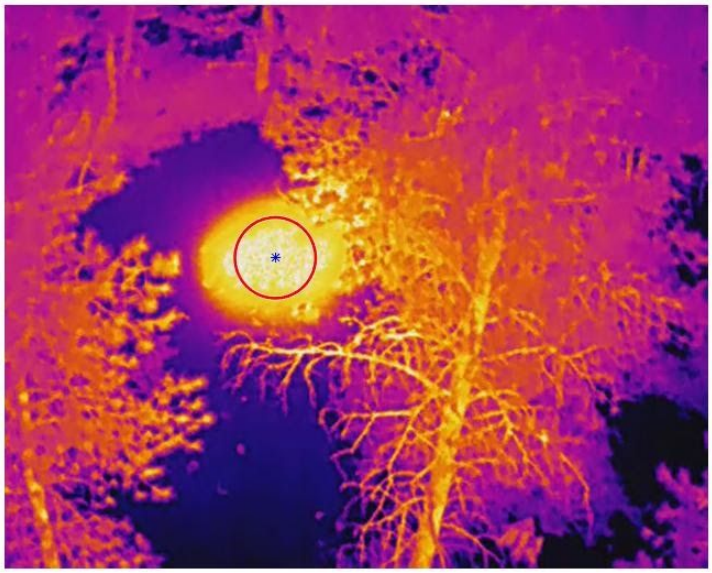


Figure 1 - Sample Raw Thermal Image

Binary Image Segmentation Result

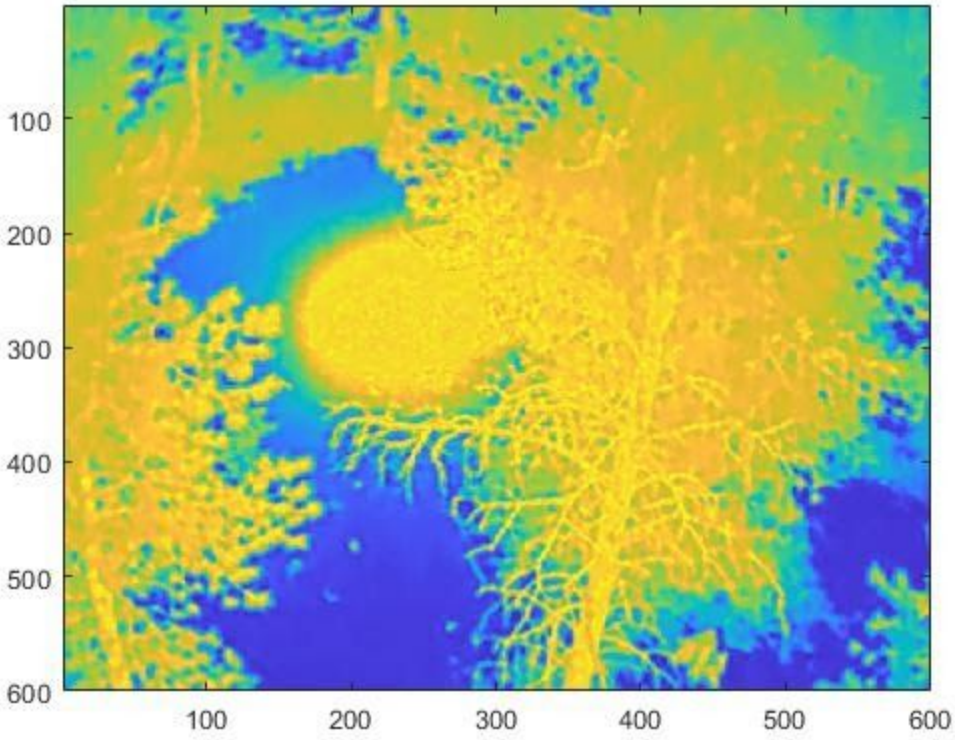
Enhanced Image

filtered image

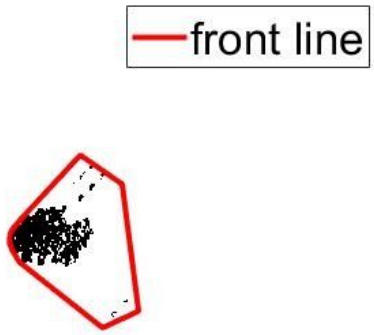


Final Detection using Binary Image Segmentation

RGB Image Segmentation Results



Separation of colors



Final fire frontline detection using RGB Segmentation

There are two reasons that our team was separated into two sub-teams. One is because of the requirements of our project. Another is that two of our teammates, Yue Zhang and Kun Xiao, know about wireless communication. Oshan and Haiyu Wu know how to do the image processing part. Based on our knowledge, we started working on our project. Fortunately, we perfectly finished our clients' requirements.

Installation of the Widget

The image processing part:

- For our simulation model:

1. Install Pytorch onto your computer
2. Load the trained model
3. Change the file name which contains the images needed to be analyzed

4. Run the file

- For our practical model:

1. Install Pytorch onto your computer
2. Load the trained model
3. Change the file name which contains the images needed to be analyzed
4. Run the file
5. Open the screen_captured.py file to capture the screen which is also the image captured by the camera
6. Run both files.

The wireless communication part:

- For our simulation module:

1. Install the GNU Radio software onto your computer
2. Open the simulated functional graph in the GNU Radio software
3. Choose what you are going to transmit and upload it onto the software
4. Attach the image file to the “File Source” block and start the simulated transmission procedure

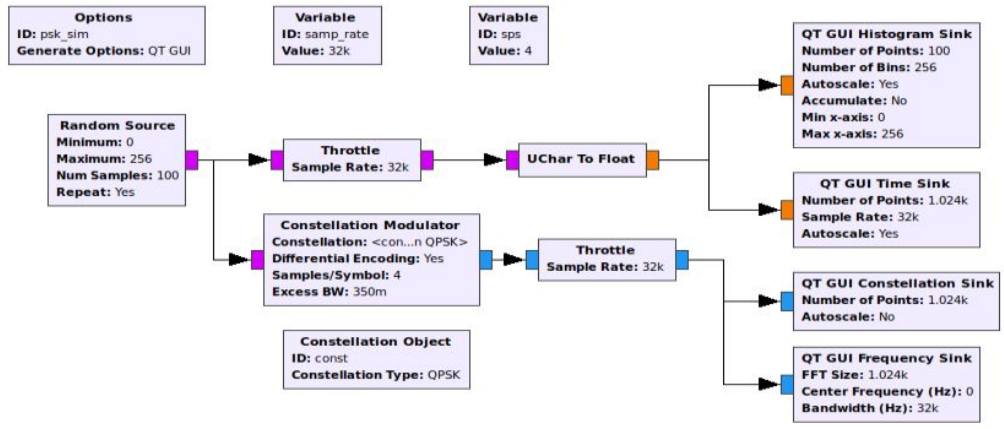
- For our practical module:

1. Install the GNU Radio software onto your computer
2. Install the USRP Driver onto your computer

3. Connect the USRP Device 1, the USRP mini, to the Raspberry Pi, download the transmitter functional sequence into it, then connect the USRP mini to the Wildfire Drone, use the mini hardware to transmit the collected image data
4. Connect the USRP Device 2, the USRP B210, to the computer, and load the receiver functional sequence into the USRP, use that sequence to receive the image data, and if needed, the user can download the image to the computer, then use the image processing module we built to analyze it and gain the required information

Configuration and Use

For wireless Communication Part:



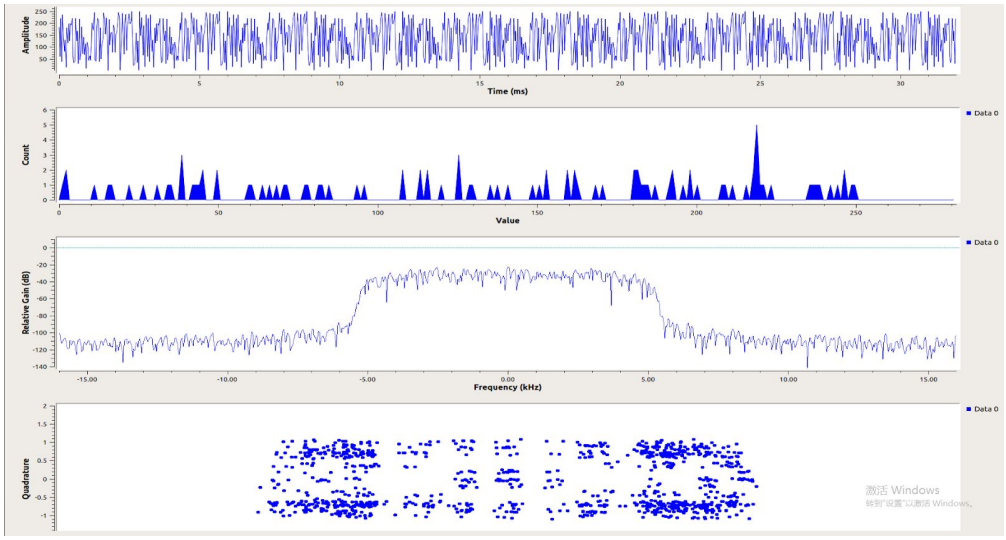
Signal Flow Graph

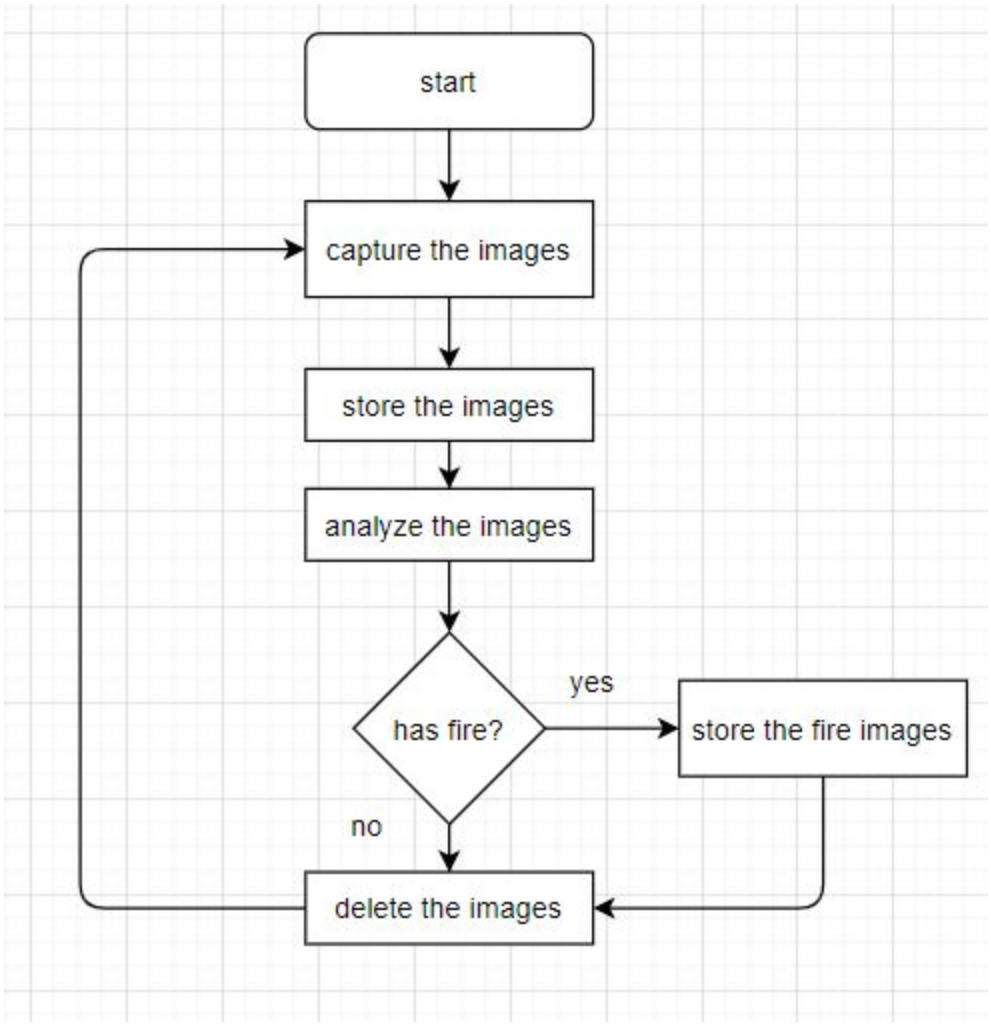
For Raspberry Pi 4, we mainly work with GNU Radio software. We use GNU Radio for a series of emulation and hardware connection USRP B205 mini. To complete the transmission of the

picture. For example, we did a simulation for the signal flow graph. A random number generator is used as input for simulation.

The source is set to repeat, creating a continuous signal that allows analysis.

The range of random number generators is set from 0 to 256. The self-output value is a byte, and a conversion block is used to change the value from an unsigned character to a floating-point value. This makes the generated signal corresponding to a readable oscilloscope. The signal is also passed to a modulation block that performs phase-shift keying (phase-shift keying). There are two signal paths, one for analyzing the input signal and one for modulating the output. Since this is an analog, the throttle block is necessary and placed in both signal paths. Four different outputs will analyze the input signal and the modulated signal. Use a histogram and oscilloscope for the input signal a. For modulated signals, the constellation, as well as the spectrum waveform, will be used. The output is displayed using a "QT GUI" processing block.





All the process will do detecting, saving, analyzing automatically as long as you start them.

Maintenance

For wireless Communication Part:

USB devices are important for using the Raspberry Pi. Whether you want to connect an input device (mouse, keyboard, monitor) or extend the computer's capabilities, many options are available via USB.

Usually, after plugging in a device, we can use it almost immediately. But what if Raspberry Pi 4 can't detect a USB device? The first thing to do is to go to eLinux.org's Verified Peripherals list. Search for USB devices on the page; Next, check that the device is running on another computer. Plug it into a PC; Is it working properly? Once you've done that, it's time to ask questions about Raspberry Pi and USB devices. On Raspberry Pi, and type in:

Sudo dmesg -c

Next, check that the device is running on another computer. Plug it into a PC; Is it working properly? Once you've done that, it's time to ask questions about Raspberry Pi and USB devices. On Raspberry Pi, type in:

Sudo dmesg -c

Next, insert the USB device and enter:

Dmesg

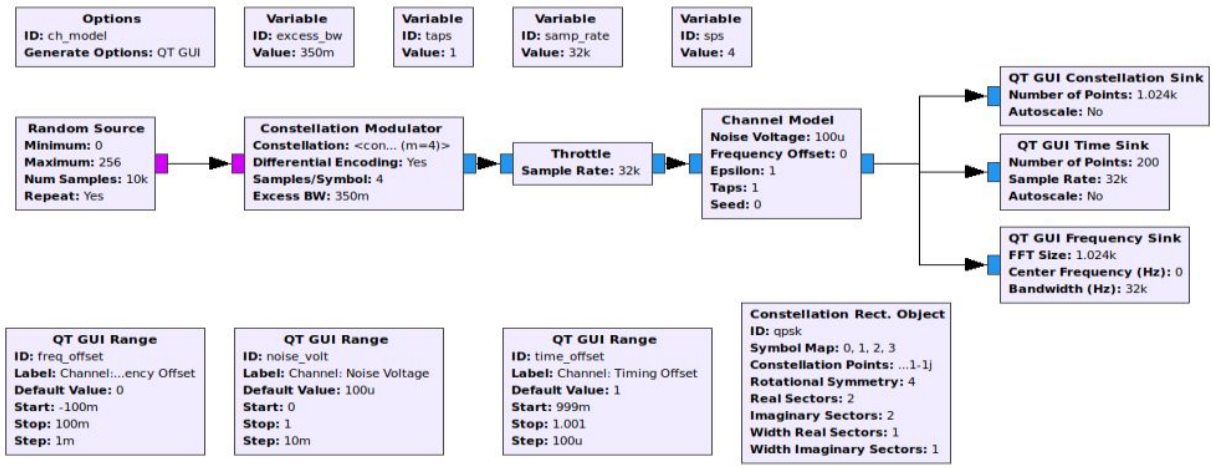
If detected, your USB device will be listed here, along with all the associated error messages. If the device is not listed, there are usually two reasons: USB device or Pi USB bus has failed. Or Raspberry Pi doesn't have enough power.

For image processing part:

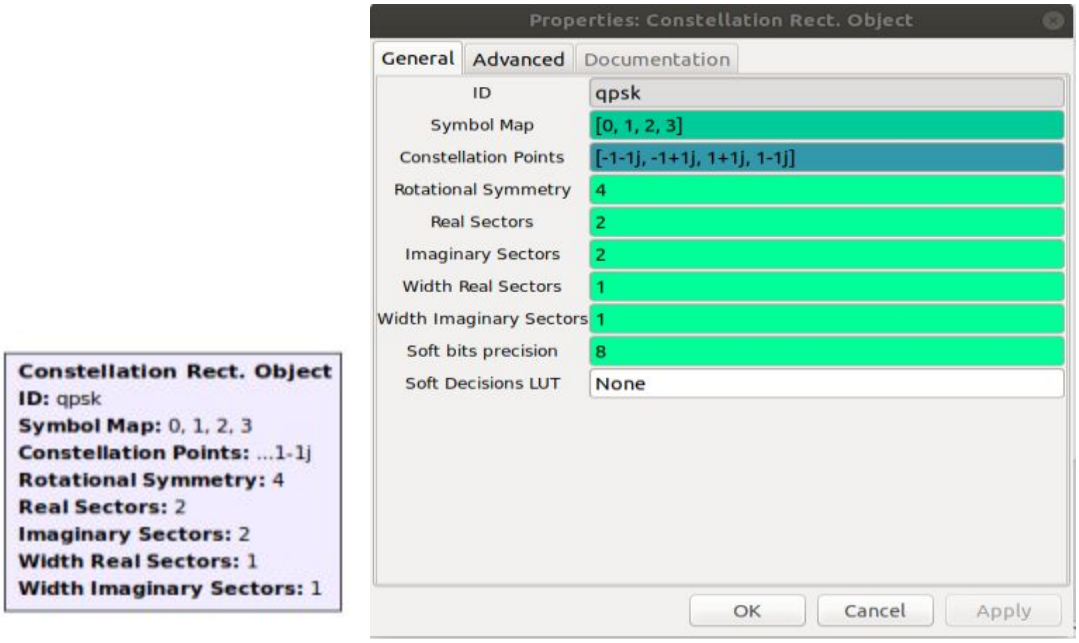
Our model could be used in any situation as long as all the tools are installed such as numpy, pytorch, etc. Another part needs to be mentioned is that if you would like to do some changes or add anything, be sure to check the file name and the file path.

Trouble-shooting Operation

For wireless Communication Part:



Simulated transmission Channel



In this simulation, when we first started using the package of Constellation Rect. Object, we used the raw date. As a result, we cannot get the result we want. Then, we changed the Constellation Points, and we got the correct result. So, we need to focus on every data of each package.

For image processing part:

1. Check if the file exist
2. Check if the file path is correct
3. Check if any tool is missed
4. Check if the trained model is in the correct file.

Conclusion

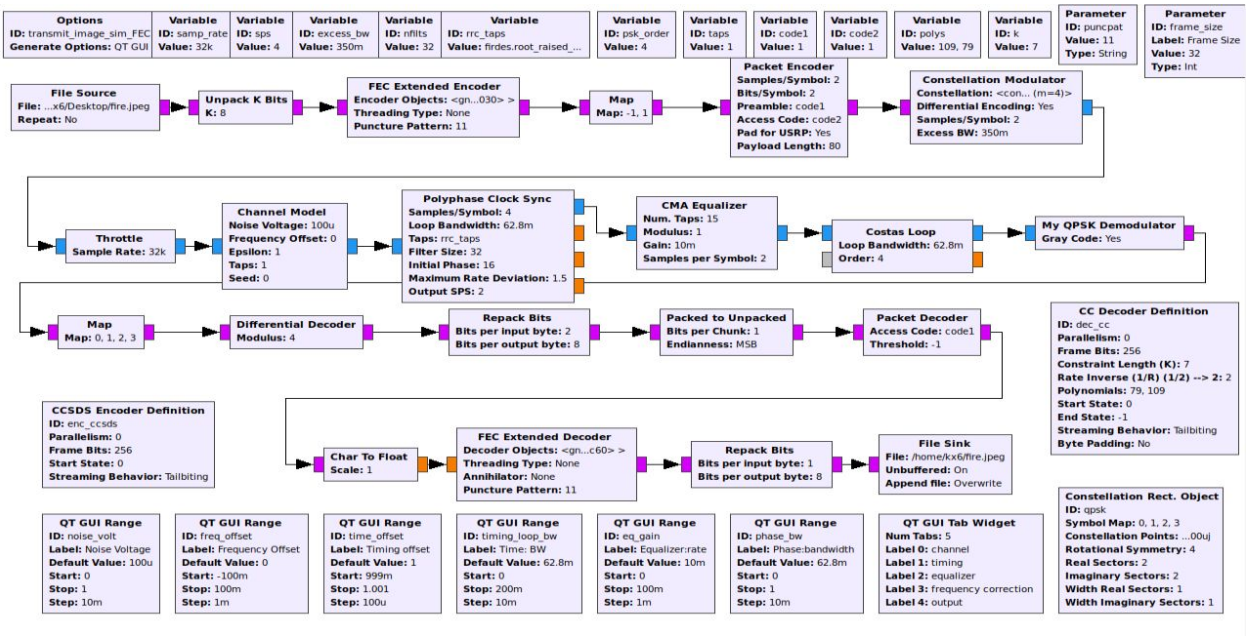
For the image processing part, our model fits all the requirements. Mobilenet allows us to install the model into the Raspberry Pi4, at the same time, it could keep the high accuracy. Moreover, because it is lightweight, the speed of analyzing is fast (0.019s per image). Our model could save the image to the specific file which could benefit the wireless communication part.

For the wireless communication part of our project, we almost finished all objectives we previously set. First, we built two functional sequences based on the USRP transmission on the GNU Radio software, one is supposed to be connected to the transmitter side, and the other one is supposed to be connected to the receiver side, and they have been tested to make a successful transmission. After the spring break, we moved to the dormitory so we began to build a simulated sequence. After we realized the simple transmitting and receiving function, we figured out there existed some error when doing the transmission. Thus, we analyzed the whole sequence and added several functional blocks to improve the reliability as well as the efficiency of the system. Finally, we finished with a functional sequence which can realize a more accurate image transmission.

Appendices with Schematics or Journal Papers

The wireless communication system:

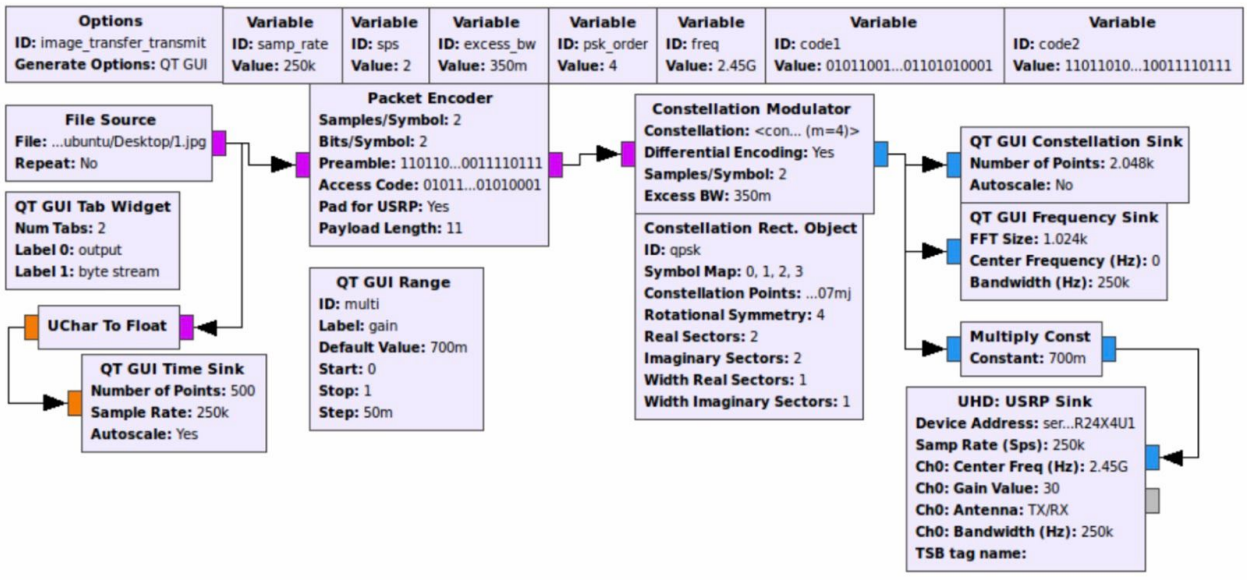
- The simulated module



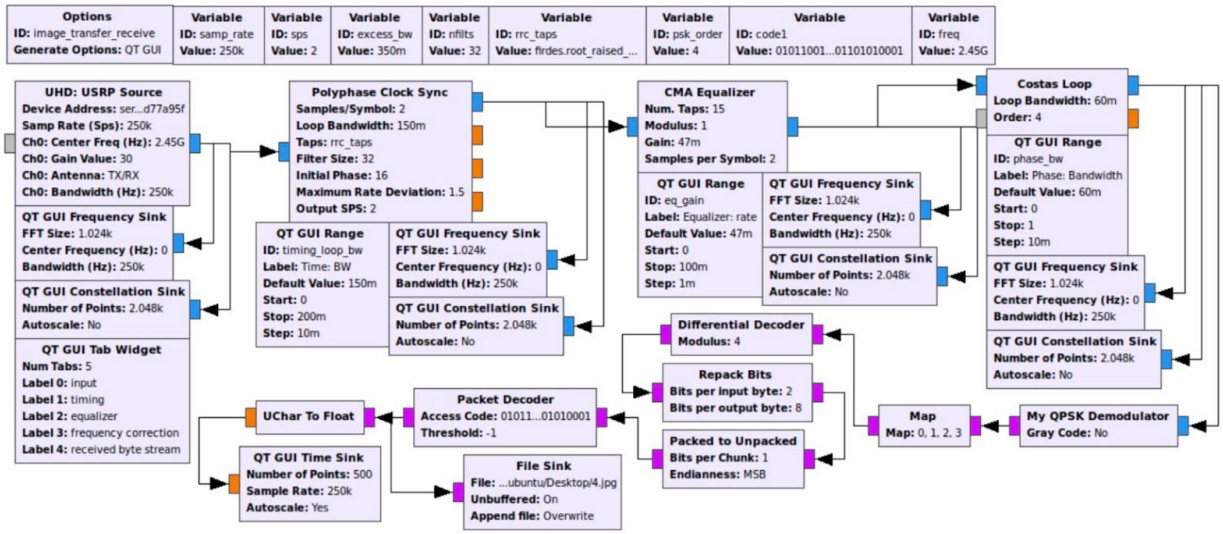
the whole simulated functional sequence

- The practical module

1. the transmitter side :

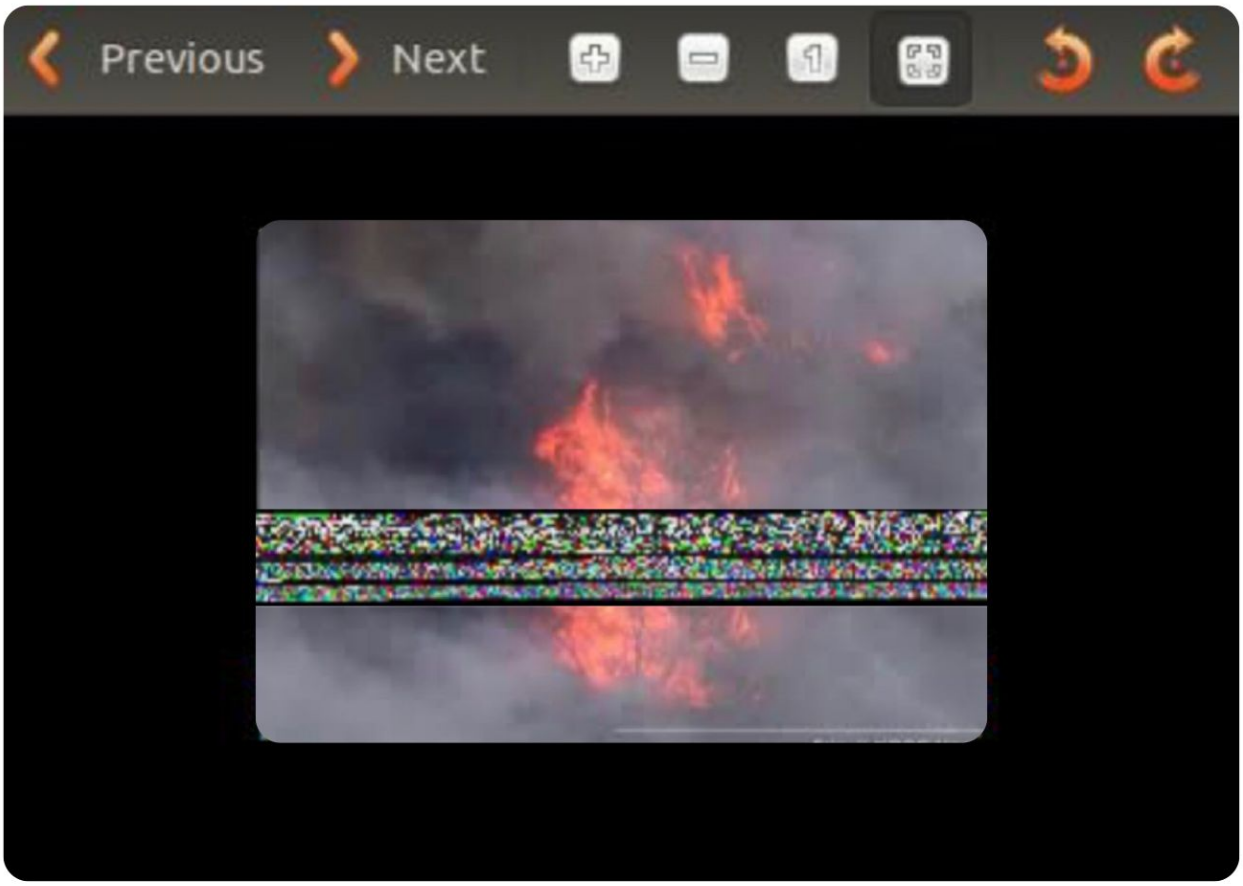


2. the receiver side :

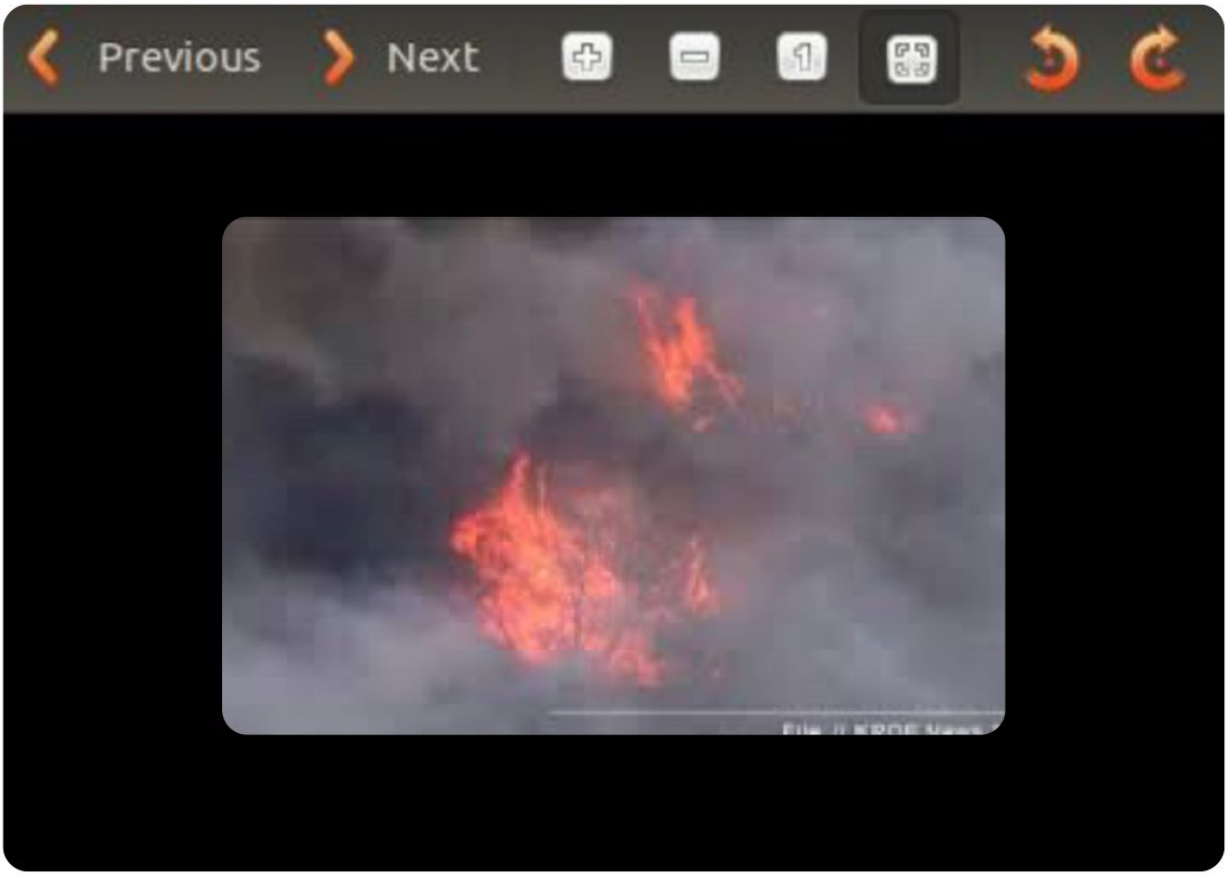


- Some of the results (testing part)

The transmitter result before the improvements:



The transmitter result after the improvements:


Previous
Next