OBJECT DETECTION AND MONITORING THROUGH UNMANNED AERIAL VEHICLE

Le Tien Thuong
Dept. Telecommunication
Ho Chi Minh University of Technology
Ho Chi Minh City, Vietnam
ThuongLe@hcmut.edu.vn

Phan Xuan Bach
Ho Chi Minh University of Technology
Ho Chi Minh City, Vietnam
1410178@hcmut.edu.vn

Nguyen Hoang An
Ho Chi Minh University of Technology
Ho Chi Minh City, Vietnam
1410167@hcmut.edu.vn

Abstract:

In this paper, we offer a solution for object detection and monitoring through unmanned aerial vehicle, specifically a drone. The drone will search, identify and follow a designated object. It can also transmit images directly to a computer, smartphone or any devices which supports RTSP protocol. It allows the observer to monitor, control, and at the same time, it can learn and remember the object temporarily so that whenever they meet, the drone can be ready to follow this object autonomously. However, the autonomous drone requires a powerful computer to do most of the recognizing object and controlling drone from its video stream, which introduces response delays on the drone.

Keywords: *Drone, object detection, unmanned aerial vehicle, digital image processing*

I. Introduction

In recent years, digital image processing has achieved many great achievements and progress. In which, image identification and classification is one of the most actively pursued research directions. Thanks to the image processing system with large amounts of training data for artificial intelligence, people have reduced a great deal of work as well as increased accuracy in making decisions related to image processing in many fields such as military and defense, biochemistry and anatomy, transportation systems, ... In those fields, security monitoring is extremely important for the daily life of people as well as very indispensable for the military and defense.

With the advancement in technology, UAVs, or unmanned aerial vehicles, are being slowly being accepted. Initially, they were used primarily for military purposes, such as reconnaissance flying with aerial photography, transmitting images to a command base, or finding and destroying difficult to reach targets with weapons. Today, in addition to specialized applications in the military, drones are also used with camera mounts (often referred to as flycams) for the purpose of aerial photography. Drone technology have developed tremendously, penetrating deeply into our lives, from entertainment, enthusiasts conquering the sky, delivering goods at a faster rate than roads, watering crops, spreading fertilizers onto crops, tracking livestock, monitoring forest for fire, illegal logging, poaching, search and rescue in dangerous places... Drones come in a variety of sizes, from compact to hold in the hand, flying in the field of view, to massive ones like real planes worth millions of dollars with ranges of hundreds of kilometers. Currently, you only need to spend a few hundred thousand to several million dong to be able to own a small drone with a controller to satisfy your joy of conquering the sky. Drones used in business activities, higher-end production, of course, are much higher in selling prices, but they promise to bring higher efficiency in use.

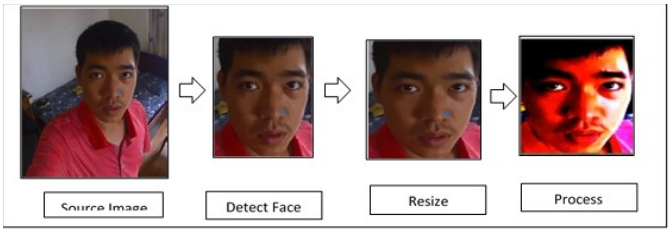
With easier access to civilian drones means is now easier to build our own drone if prebuilt drones on the market can't meet our expectations. We present a drone built from scratch, using materials easily found, to recognize a person specified in training dataset and follow that person. At the heart of the drone is the flight controller and the Raspberry Pi + Arduino Uno to transmit video capture to and receive command from the ground station (a.k.a a computer). The Arduino Uno acts as intermediate between the Pi and what the flight controller to control the drone.

II. Object detection and tracking algorithms

1. Face recognition

1.1. Facenet

A Convolution Neural Network (CNN) is a neural network formed from convolutional layers that perform the characterization of an image. Result from the image is then transferred to another machine learning model such as kNN or SVM to distinguish this person with others. This implementation is called one-shot learning. CNN only plays the role of feature extraction, and training CNN here is to make it better characterization. Facenet is essentially a CNN that separates the features of a face from an image. What makes Facenet special about it is that it uses the Triplet error function to minimize the distance between similar faces and maximize the distance to dissimilar faces, which help Facenet distinguish between person to person accurately. The CNN training is performed on large datasets (e.g vggface or MS 1 million), the result we want is that the embedding vector, which comes out as a result of CNN, has to be split into clusters to perform classification by kNN or SVM.



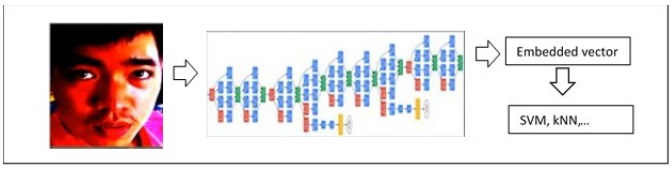


Figure 1: Facenet preprocessing.

Figure 2: Inner working of Facenet.

Use a face detector to separate part of an image with a human face. Modern methods use MTCNN (Multi-task Cascaded Convolutional Networks), the advantage is that this model identifies faces in many angles, and can recognize face boundaries with quite good processing time. However, we found out another method that decrease processing while still maintaining similar results is using SSD (Single Shot Detector). Then we resize the image to a specified size, the Facenet requires 160x160 [1].

1.2. Single Shot Detector [5]

SSD is designed to detect objects in real time. SSD speeds up the process by eliminating the need for region proposal networks. To address the reduced accuracy issue, SSD applies a few enhancements including multi-size feature maps and the use of anchor boxes. These enhancements allow the SSD to come close to the accuracy of Faster R-CNN but be able to use lower resolution images, resulting in higher speeds.

The SSD model is divided into two phases:

1. Extract feature map (based on VGG16-based network) to increase efficiency in detection.

2. Apply convolutional filters.

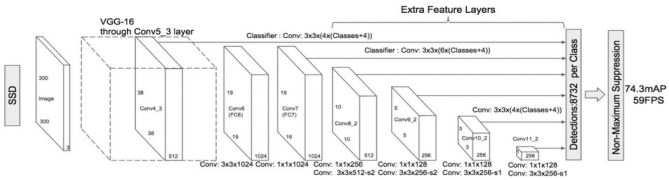


Figure 3: SSD model.

We use a pretrained Tensorflow version of ResNet10-based SSD from OpenCV Github page for face detection [6]. It is then loaded by “dnn” module from OpenCV library to detect faces and scale back to 160x160.

2. Object tracking

2.1. YOLO (You only look once)

YOLO was developed by Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi [2] in 2015. As a CNN model to detect objects, YOLO has the remarkable advantage of being much faster than other models, even running on devices like the Raspberry Pi [3].

There are 4 advantages of YOLO [4]:

- Fast speed. Because the object detection and coordinate locator system has been combined into a CNN network, it is possible to feed images and even video for identification. Compared to Fast R-CNN, in which each run will generate 2000 position predictions, YOLO is hundreds of times faster.
- Can "see" the whole photo compared to each part like R-CNN, from which its predictions are provided with the global content of the photo. Compared to Fast R-CNN, YOLO only makes less than half of recognition errors.
- Compact. YOLO only uses a network that goes directly from the convolutional layers to the Fully connected layers. The image is only passed once and then it returns the output. That is why we call it the "only look once" algorithm.
- Open source. We can change and improve as we like.

For our proposal, we use YOLOv3-tiny for increased accuracy while keeping processing time at a minimum.

2.2. Handling occlusion

For object detecion and tracking, what should be kept in mind is occlusion. "Occlusion" here means obscuring each other in movements results in the subject being tracked disappearing and then reappearing, or two objects overlap leading to indistinguishability. Both of the above cases result in the disappearance of the tracked subject.

3. Video Streaming

Streaming video is a technique used quite commonly in network applications. A lot of sites widely used streaming in practice such as: software (media player, web browser, ...) on the client computers to access and watch videos from servers in the server / client model; online meeting and distance training applications; monitoring, remote control via real-time images, etc.

Streaming video uses a way of replaying videos stored in computers on the network to end users who want to watch the video without downloading the video on their computer. In essence, streaming video is the process of splitting a video file into frames, then sending each frame to a buffer on the viewer's computer and displaying the frame's content. And this process strictly adheres to time constraints, in other words the need to adhere strictly to real-time transmission protocols such as RTSP, RTP and RTCP. With such a feature, streaming video is a fairly complex technique to deploy. However, this is a must for the proposal due to the need to track the subject on the computer screen.

Since it is necessary to stream video to the computer for image processing and automatic flight control, it is important to note here that the processing speed must be as high as possible to achieve the best result for the topic.

III. Drone

1. Overview

Unmanned Aerial Vehicle or UAV (Unmanned Aerial Vehicle) is a common name for aircraft without a pilot in the cockpit, operating independently and often remote-controlled from center or controller. A UAV is defined as an unmanned, airborne vehicle that uses aerodynamics to provide lift, can fly autonomously or remotely, can be recovered or not, payload or not. Consequently, the missile is not considered a UAV as it is used as a weapon and not a means of transportation, and cannot be recovered for reuse, although it is also unmanned and some can be controlled remotely.

2. Common structure of a drone

Basically, a drone will have the following components:

- RC controller: This controller theoretically, when there is no obstacle, can control within a radius of 1km. To control the UAV requires 4 or more channels. The joysticks include left, right, up, down, left rotation, right rotation, self-balance...
- Main control board (or flight controller): This board is responsible for receiving control signals from the joystick, reading the balance sensor data on its body to control four motors at 4 corners accordingly. The balance sensor on this board is very useful because it helps to adjust the drone when just taking off. Modern boards also integrate GPS on it to help locate the plane or land automatically.
- Drone frame: This frame is designed so that the 4 corners are completely balanced in both weight and size. Motors located at these four corners must operate synchronously. When we assemble the frame we need to ensure its symmetry.
- Motor and pulse controller: After the controller board receives signal from joystick and generates control pulses to the flight controller, the pulse are generated according to the pulse width and control the motor speed. The engine is a one-way motor with high rpm, can be up to 750 ~ 1000 rpm. Current consumption can go up to 35A. This is also the reason why these types of aircraft can only fly for 10 ~ 15 minutes because the power consumption in the engines is very large.
- Wing or fan blade: This is extremely easy to damage if you do not control it well, causing the aircraft to crash. Aircraft wings are usually made of plastic or carbon fiber. You should use a carbon fiber wing because it is extremely well-balanced in design and very high rigidity will help reduce damage when the plane crashes.
- Power source: LiPo battery with current output up to 30A.

To be specific, this proposal uses components manufactured by NAZA, the NAZA M-LITE kit:

- Main Controller (MC): The Main Controller (MC) is the brain of the system, it communicates with all ESCs and RC transmitter to carry out the autopilot functionality. It has a built-in Inertial Measurement Unit (IMU) consists of one 3-axis accelerometer, one 3-axis gyroscope and a barometer for sensing the attitude and altitude. (There is a software to support the parameter setting for flight systems developed by NAZA).
- Versatile Unit (VU): Specially designed for NAZA-M LITE. It solves the high power consumption problem of multi-rotor system, supply and monitor power for NAZA-M LITE and other electronic devices. It also has an LED to indicate different operating states of NAZA-M LITE and a USB interface for configuring the NAZA-M LITE unit and firmware upgrade.
- Optional GPS & Compass Module: The GPS/Compass module is for sensing the position and direction. Because the GPS & Compass are sensitive to magnetic interference, you should use this bracket to mount the GPS module.
- Engine: Use 4 Sunnysky A2212 - 980KV brushless motors, 3S-11.1V LiPo working voltage. Accordingly, the propellers match the motors.

- The frame: TAROT IRON MAN 650 uses carbon fiber material, which is both flexible and rigid for experimentation and easy to replace if there is any damage.
- RX701 and Devo 7: DEVO 7 controller is a 6-channel controller, single remote controller without receiver circuit, can connect to RX 601, 701, 801 or RX1002 receiver (model current is associated with the receiver RX701). DEVO 7 Controller Specifications:
    + Encoder: ARM microcontroller
    + Frequency: 2.4G (DSSS)
    + Output power: ≤100mW
    + Control signal range: 200 – 400m (depend on RX)
    + Current consumption: ≤220mA
    + Battery: 1.2V*8 pin nickel-cadmium hoặc1.5V*1 pin 8AA
    + Output width: 900-2100mS (1500mS center)
    + Size: 190 * 178 * 67 mm
    + Weight: 652 gram

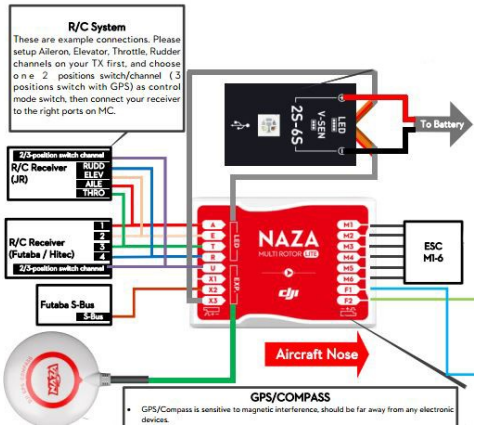


Figure 4: How to connect components according to the manufacturer's instructions

3. The theory of UAV flying behaviors

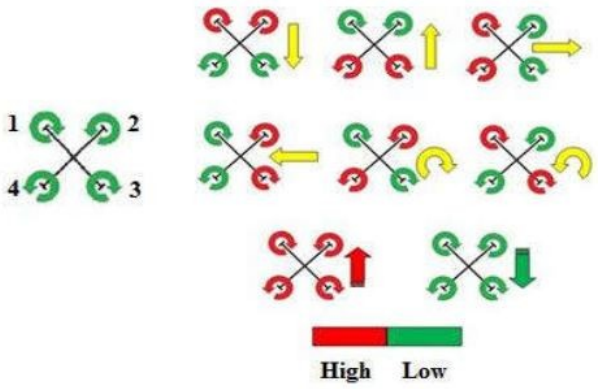


Figure 5: Principles of movements of quacopter.

In 4 engines, 2 opposite motors will reverse with the other 2 to create torque, helping the device to fly. This can be described as shown below.

Regarding the UAV's flight behavior, we can divide it into three basic directions:

- Altitude: Here, the plane will fly for altitude adjustment, that is, fly high, or fly down low. To do this, one can either increase or decrease the thrust of all 4 engines.
- Deviation: We can control the flying device to the left or right by increasing the thrust on one or several engines to adjust them to fly at will.
- Rotation: The drone can be rotated in place by increasing the thrust in one engine while reducing the thrust in the opposite engine.
- The proper and skillful coordination of these movements of the device makes the flight behavior of the device smoother, making it easy to record images.

IV. Implementation

1. Model overview

Based on studying the problems posed and in turn solving these problems, we intend to create a basic model as shown below

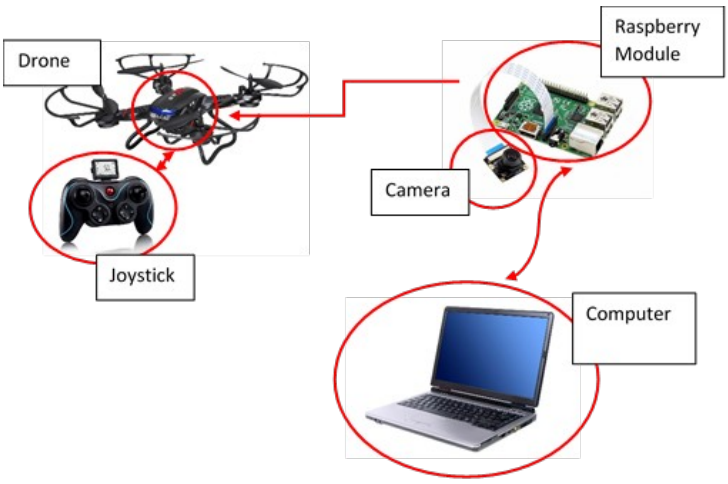


Figure 6: Diagram of system planned for implementation

Table 1: Drone components

Components	*Specifications*
Drone frame	Tarot Iron man 650
Flight controller	Naza M-lite
Engine	980 KV
RC Receiver	RX701
LiPo Battery	5200 mAh
Fan blade	Emax 10x4.5
Camera	Raspberry Pi Camera Module V2-5 Megapixel,1080p
CPU	Raspberry Pi 3 + Arduino Uno
Connectivity	Wireless (Wifi)

The system will have 2 modes of operation: manual control mode, or automatic search and monitoring mode.

First, a flying device with a main controller connected to the Raspberry Pi Module is equipped with the module's dedicated camera. The Raspberry Pi will be responsible for recording and sending the image captured from the camera to the computer for the computer to process, identify the object and send the control signal back to the Raspberry Pi. Thus, Raspberry Pi is considered as the signal transmission module between the drone and the computer. From there, Arduino Uno takes the control signal received by the Rasperry Pi and translates it to signal pulses that the flight controller can understand and controls the drone.

From the computer, the user interface is used to control the aircraft manually or switch to automatic search mode.

The joystick is always ready to connect to the flying device using the switch on the device in case the signal connected to the computer is disconnected or the control program hangs, causing the drone to lose control. From here, failsafe mode, which is in the flight controller independent from the Pi + Arduino Uno, can be activated to ensure the drone lands safely.

2. Interfering signal transmission between the RX701 and the flight controller

The original concept for the model was to use a Raspberry Pi to handle everything, including video transmission, image processing, and flight control. However, there are two problems that prevent this. Firstly, if using only Raspberry Pi, the workload is too large leading to slow processing speed; Second, the operating voltage levels of the RX701 and the Pi ports differ, resulting in difficulty in signaling. Therefore, the proposed solution is to assign the signal relay (in manual mode), fake the signal from the controller to control flight behavior, or choose the operation mode for the model. for Arduino Uno kit. The Raspberry Pi will only be responsible for transmitting the image to the computer, receiving the command after processing is done, forwarding it to the Arduino and then to the flight controller.

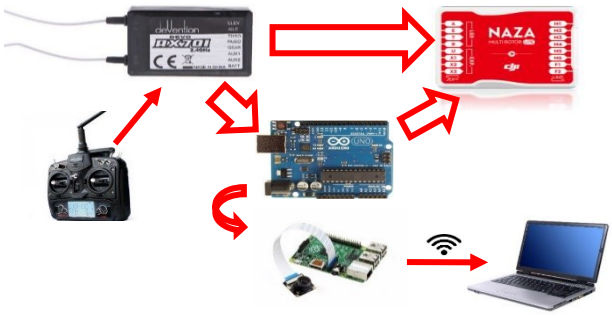


Figure 7: Interfering signal transmission

Get started with reading the RX701's signal pins using a joystick. The better we understand how it works, the easier it is to fake these signals for automatic control at will.

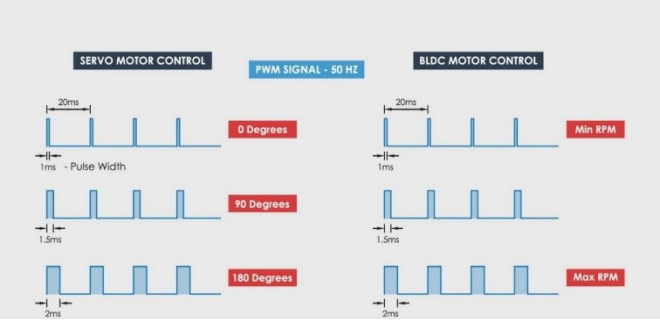


Figure 8: PWM signal for servo is similar to what the receiver put out

The joystick and motor control signals both use PWM control signals. Joystick basically consists of 4 channels corresponding to 2 levers, each side can be pushed up or down or left and right, corresponding to the increase and decrease for the channels. The left lever controls 2 channels that make the drone fly up, down and rotate left and right; the right lever controls the remaining 2 channels flying forward, backward and left and right. There are also buttons to control other channels such as Gear, Mux, ... we can use these channels to switch the flight mode for the model (manual or automatic).

Measurement of the channel values obtained the following results:

Left stick:

When pushed highest: THOR on RX701 gives PWM 1900 us (or 1.9 ms)

When pulled lowest: THOR on RX701 gives PWM 1100 us (or 1.1 ms)

When pushed full left: RUDD on RX701 gives PWM 1900 us (or 1.9 ms)

When pushed full right: RUDD on RX701 gives PWM 1100 us (or 1.1 ms)

Right stick:

When pushed highest: ELEV on RX701 gives PWM 1900 us (or 1.9 ms)

When pulled lowest: ELEV on RX701 gives PWM 1100 us (or 1.1 ms)

When pushed full left: AILE on RX701 gives PWM 1900 us (or 1.9 ms)

When pushed full right: AILE on RX701 gives PWM 1100 us (or 1.1 ms)

The shift:

GEAR has 3 levels: 1860 us at highest position, 1384 us in the middle, 1192 at bottom position.

AUX1: 1900 us, 1500 us, 1100 us.

Mux: same as AUX2: highest position 1100 us (1.1 ms), lowest position 1900 us (1.9 ms)

From these values, it is possible to simulate the drone's flight behavior by coordinating the respective channels.

3. Real time video streaming.

3.1. FFMPEG library

Raspberry Pi Camera is a module specially produced for the Raspberry Pi series, with different versions, different recording speeds and image quality. Depending on the purpose of use can be selected accordingly. The basic OpenCV video streaming library is FFMPEG.

Raspberry Pi Camera can output video in 2 formats: MJPEG and H264. Each format has its own characteristics.

+ MJPEG has good image quality, high capacity, very bandwidth consuming transmission, video stream with http protocol, real-time standard.

Use the raspivid program to get the image from the camera in MJPEG format, then push through the VLC to transmit over the http protocol:

raspivid -n -cd MJPEG -t 0 -v -w 1280 -h 720 -fps 30 -b 8000000 -o - | cvlc stream:///dev/stdin --sout '#standard{access=http, mux=mpjpeg, dst=:8081}'

On PC we need to input to OpenCV: 'http://Pi_address:8081/'

However, the weakness of this approach is the need for very high bandwidth, 8 Mbps (Megabits / s). When the drone is flying high, the image is jerky and stalled due to the inability to transmit, even though the transmission on the ground is achieved in real time.

+ H264 reduced image quality, low capacity, fast transmission speed, transmission via http or rtsp protocol, delay made it impossible for real-time transmission.

Use the raspivid program to get the image from the camera in H264 format, then push it through VLC via http or rtsp protocol:

RTSP: raspivid -n -t 0 -v -w 1280 -h 720 -fps 30 -b 2000000 -o - | cvlc -vvv stream:///dev/stdin --sout '#rtp{sdp=rtsp://:8081/}' --demux=h264 --h264-fps=30

HTTP: raspivid -n -t 0 -v -w 1280 -h 720 -fps 30 -b 2000000 -o - | cvlc -vvv stream:///dev/stdin --sout '#standard{access=http, mux=ts, dst=:8081}' --demux=h264 --h264-fps=30

On PC we need to input to OpenCV: 'http:// Pi_address:8081/' hay 'rtsp:// Pi_address:8081'

This reduces the bandwidth required for video transmission. The weakness of this approach, however, is the inability to achieve a real-time response, possibly because VLC cannot transmit fast enough.

3.2. Gstreamer library

GStreamer is a pipeline-based multimedia framework that links together a wide variety of media processing systems to complete complex workflows. For instance, GStreamer can be used to build a system that reads files in one format, processes them, and exports them in another.

We switch to the GStreamer library to transfer the images from the drone to the PC. Raspivid will be passed through Gstreamer to create a server on the Pi to be connected and receive images. It is possible to adopt 3 protocols: TCP, UDP or RTSP to transmit video.

a) Through TCP

You can use two basic lines of command to perform video transmission with the Gstreamer library as follows:

raspivid -n -t 0 -v -w 1280 -h 720 -fps 30 -b 2000000 -o - | gst-launch-1.0 -v fdsrc ! h264parse ! rtph264pay config-interval=1 pt=96 ! gdppay ! tcpserversink host=0.0.0.0 port=8081

Through this command line, GStremaer will create a connection to Pi to watch the video:

gst-launch-1.0 tcpclientsrc host=< Pi_address> port=8081 ! gdpdepay ! rtph264depay ! avdec_h264 ! videoconvert ! autovideosink sync=false

In this way, the video transmitted is not delayed as much as before..

b) Through UDP

raspivid -n -rot 180 -t 0 -a 512 -v -w $width -h $height -fps $fps -b $br -o - | gst-launch-1.0 -v fdsrc ! h264parse ! rtph264pay config-interval=1 pt=96 ! udpsink host==<receiver_IP> port=8081

Through this command line, GStremaer will create a connection to Pi to watch the video:

gst-launch-1.0 udpsrc port=8081 ! rtph264depay ! decodebin ! autovideosink sync=false

With UDP transmission protocol, video transmits a bit faster than TCP protocol, reduces delay a lot, can be considered as parallel transmission in real time. However, this requires knowing the address of the receiving PC, unlike how TCP will create a server to connect to.

Either way, there is no way for OpenCV to get frames from GStreamer. Therefore, it is necessary to recompile OpenCV to support both FFMPEG and GStreamer.

c) Through RTSP.

Enter this line to activate the RTSP server:

/home/pi/gst-rtsp-server-1.14.4/examples/test-launch --gst-debug=3 "(rpicamsrc bitrate=2000000 preview=false ! video/x-h264, width=1280, height=720, framerate=30/1 ! h264parse ! rtph264pay name=pay0 pt=96)"

Then with OpenCV we have access to FFMPEG as before: 'rtsp:// Pi_address:8554/test'; Or use the GStreamer library:

rtspsrc location=rtsp://<Pi_address>:8554/test latency=0 buffer-mode=auto ! decodebin ! videoconvert ! appsink sync=false

Through RTSP from GStreamer, we make video stream in real time.

We also tried the following command:

gst-launch-1.0 -v rpicamsrc bitrate = 2000000 preview = false! video / x-h264, width = 1280, height = 720, framerate = 30/1! h264parse! rtph264pay config-interval = 1 pt = 96! gdppay! tcpserversink host = 0.0.0.0 port = 8081

However the image that PC receives is always damaged, broken.

4. Algorithm flowchart

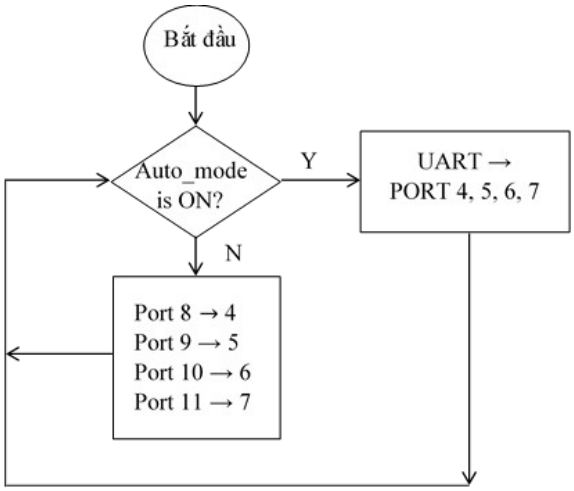


Figure 9: Signal forwarding

In control forwarding, we need to consider which mode is the current flight mode. With manual flight mode, Auto_mode off, RX701 receives control signal from DEVO 7 and then transmits to Arduino, Arduino will only forward it to flight controller to control drone. With the auto flight mode, Auto_mode turned on, the Arduino stops receiving the signals of the 4 main control channels from the RX701. Instead, it translates the code received at UART (or USB) from the Raspberry Pi and then transmit it to the PORT 4, 5, 6, 7 to the flight controller to control flight.

The data forwarding was not really as stable as expected. The initial signals from the control directly connected to the flight control circuit have basic stability so that no flight problems occur. However, the Arduino working voltage level for the PWM signals received in real time as well as the PWM pulse generation at PORT 4, 5, 6, 7 for flight control is not really stable. When the USB is connected to the flight controller and use the included drone program on PC to check for received data at control input, the numbers that shows the pulse level fluctuate over a much larger range than the actual control signal. Basically this pulse can still control the aircraft, however, there will be occasional cases of unstable flight due to this.

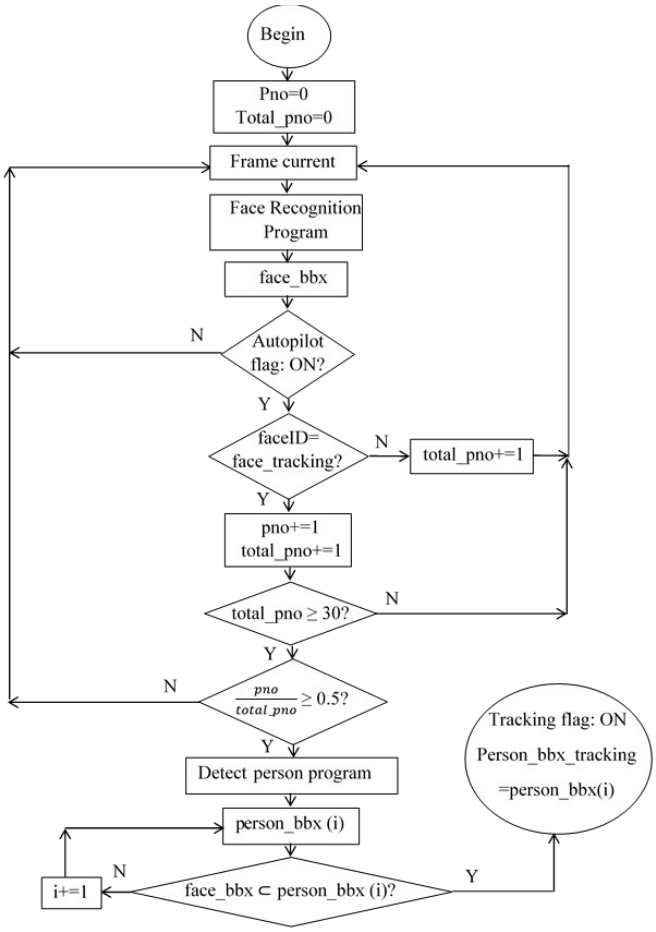


Figure 10: Algorithm flowchart.

Object recognition and flight control are two core issues of this topic. In object recognition, there are two things that need to be done. First of all, the first thing to do is face recognition, see if this subject is something to be tracked or not. The FaceNet network will work with a trained face, compare and generate an ID for this object. Recognition has a fairly high probability of accuracy, however, in some cases misidentification still occurs, so it is necessary to sample in a number of processed frames and then calculate the ratio between recognized frames to total frames since the recognized subject's first frame detected. If the recognition result is correct and the subject's probability is higher than the set threshold (here is 50% of the processed frame) then move on to the second step, identify the person. The YOLOv3 network will be started and will present bounding boxes that delineate objects identified as "people". From the results of FaceNet, we consider whether the coordinates of the recognized face are in the restricted box or not. If so, lock the subject and start tracking.

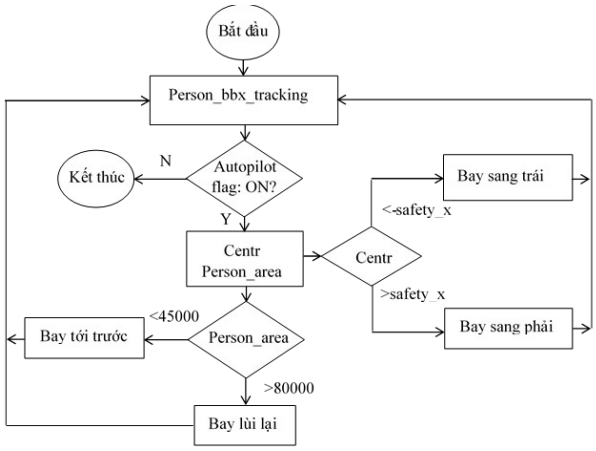


Figure 11: Flight control flowchart.

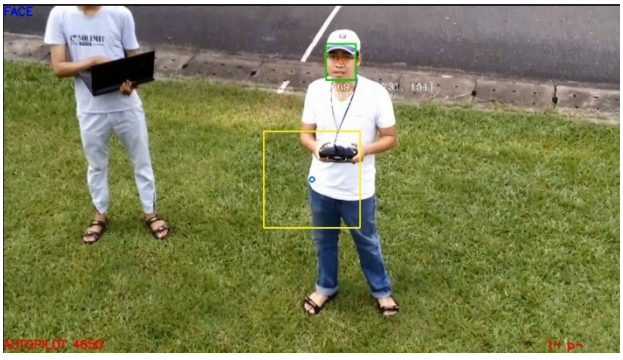
Flight control has always been the most difficult issue identified since the beginning of this topic. The direction of implementing the controls is not really perfect, but the results are still quite satisfactory.

First of all, we have to define a safety zone with the aim of keeping the subject's image in our frame at all times, while keeping the person under surveillance safe as well as the flight device (approximately this is supposed to be in the middle of the frame). From the bounding box coordinates of the person being tracked, we compare it with the coordinates of the safety zone, combined with the person's bounding box area (this is related to image processing with depth). Then we can proceed for Arduino board to generate PWM pulses sent to the PORTs of the flight according to the requirements of the proposal.

V. Performance results and conclusions

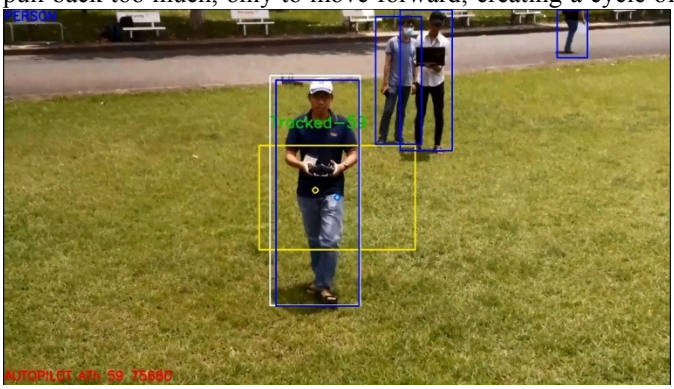
5.1. Results

First of all, in object recognition, facial recognition as well as human recognition models applied to the topic gave quite satisfactory results. Processing speed as well as accuracy are quite high, although this still depends a lot on environmental conditions (ambient light, sunlight, distance, ...) which is a general problem when processing moving images by camera. In particular, the distance to the person being tracked is really a difficult problem. If too far, the camera gives too blurred of an image, the aspect ratio of the face is too small, the neural network cannot recognize the face. If it's too close, it's really not safe. Not to mention what could affect a supervised person, just the stability of the flight system and the loss of control (if possible) are both dangerous for both the person and the aircraft (although there are contingencies available for these cases). The transition from facial recognition to human body recognition still has a small transition problem, it involves the deployment of the neural network, the network booting burdens the computer and makes the process stall in a moment. However, it does not have a major effect on the whole process.



In the process of drone flying after a tracked subject, although only using 2/4 control channels, the results are still very good. The view from the camera always closely follow the monitored subject with only 4 actions: forward, backward, left rotation, right rotation.

The problem here is that the deep learning model's sensitivity is not really good and the processing is still delayed compared to the desired process, thus leading to drifting inertia. For example, when we need to move forward (because the area of the human limit box is less than the set threshold) the fly forward command is transmitted. However, when a situation occurs, the image is sent back to the computer. The command from the computer is sent to the Raspberry Pi + Arduino, which is translated into a motor control pulse. This whole process is slower than expected, resulting in the aircraft moving forward more slowly than necessary; The flight inertia of the aircraft is also not handled well by the main flight control board, so the flight usually lasts longer than expected (instead of flying to 20cm ahead, it will float another 3-5cm). This causes the aircraft to exceed the threshold setting and the aircraft has to reverse again; pull back too much, only to move forward; creating a cycle of going back and forth that is difficult to restrain. This is really an error



that needs to be improved to make the proposal more complete.

5.2. Reviews and conclusions

The results of the proposal are not really perfect, but basically it has satisfied the requirements set forth.

A few disadvantages needed to be overcome are as follows:

- Laptop configuration requirements are quite high, delay in signal transmission as well as poor processing speed will directly affect model results.
- The performance of the computer (between using direct power source and using battery) is also important issue when implementing the topic. Because when using battery, the computer goes into power saving mode, not using the GPU at maximum limit. It is best to use power directly from the power network to prevent this problem.
- The distance to detect faces is quite limited, face detection models are highly dependent on the face size, face angle as well as the brightness of the face. At the same time, improved facial recognition accuracy will speed up processing for drone control.
- The power source for the flying device is limited (each battery only has about 5200mAh equivalent to 10-15 minutes of flight time).
- The distance of the wifi signal is quite small, limited to a certain campus; The flying device is quite large and cumbersome, so the topic is limited to outdoors, open space, few obstacles, difficult to deploy in a room.

VI. Future Developments

This is a fairly new topic, so there are many possible developments in the future.

- Since the wifi signal from the router is limited in a specific space, if you want to track objects that are far away or very far away, it is not possible. Therefore, if it is possible to switch the wifi network to a mobile data network, transfer directly to the web server then wherever we are, we can still control the object tracking as desired..
- The power source is quite limited, so if it's better to increase the battery capacity.
- The size of this drone is quite large (in return for stability in strong winds or adverse weather). If it is possible to shrink the size and still ensure a strong structure that can be stable in adverse weather, it will be a lot more convenient for object tracking as desired.

- The specific tools to implement the topic seem quite cumbersome, so if you integrate the main flight controller board, Arduino, Raspberry Pi, ... into a single device, you can greatly reduce the weight of the device. flight, thereby increasing the flight time of the device as well as its flight stability.

References

[1] Florian Schroff, Dmitry Kalenichenko, James Philbin, “FaceNet: A Unified Embedding for Face Recognition and Clustering”, IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2015, 2015, ISBN *978-1-4673-6964-0*

[2] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, “You Only Look Once: Unified, Real-Time Object Detection”, Conference Proceedings of the IEEE conference on computer vision and pattern recognition, p.779-788, 2016, ISBN *978-1-4673-8851-1*

[3] Quoc Pham, “Tìm Hiểu Mô Hình YOLO Cho Bài Toán Object Detection - Understanding YOLO”, https://pbcquoc.github.io/yolo, 2018.

[4] Long Đỗ, “YOLO: You Only Look Once”, https://ai.hblab.vn/2017/10/yolo-you-only-look-once.html, 2017.

[5] Nguyễn Tuấn Anh, “Detect object sử dụng mô hình SSD”, https://viblo.asia/p/detect-object-su-dung-mo-hinh-ssd-WAyK84rnKxX, 2019.

[6] “OpenCV: Open Source Computer Vision Library”, https://github.com/opencv/opencv/tree/ master/samples/dnn/face_detector

[7] Stephanie Stocker, "6 Degrees of Freedom Graphic", https://www.ceva-dsp.com/ourblog/what-is-an-imu-sensor/hil-what-is-an-imu-wk-1/, 2019.

[8] Pietra F T Madio, "A FaceNet-Style Approach to Facial Recognition on the Google Coral Development board", https://towardsdatascience.com/a-facenet-style-approach-to-facial-recognition-dc0944efe8d1, 2019.

[9] Joseph Redmon, Ali Farhadi, The Deal, “YOLOv3: An Incremental Improvement, University of Washington”, 2018, pp. 1-3.

[10] Craig Whitlock, “Part One: War Zones”, When drones fall from the sky, https://www.washingtonpost.com/sf/investigative/2014/06/20/when-drones-fall-from-the-sky/, 2014.

[11] Federal Aviation Administration, “Fact Sheet – Small Unmanned Aircraft Regulations (Part 107)”, https://www.faa.gov/news/fact_sheets/news_story.cfm?newsId=22615, 2018.

[12] Rituparna Chatterjee, “How image processing will change your world in future”, https://economictimes.indiatimes.com/tech/software/how-image-processing-will-change-your-world-in-future/articleshow/ 10394958.cms, 2011.

[13] Andrew G. Howard, Menglong Zhu và các cộng sự, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications”, Google Inc., 2017.

[14] Xingyi Zhou, Vladlen Koltun, Philipp Krähenbühl, “Tracking Objects as Points”, IEEE, 2020.

[15] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, Yu Qiao, “Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks”, IEEE, 2016.

[16] Jiankang Deng, Jia Guo, Niannan Xue, Stefanos Zafeiriou, “Jiankang Deng, Jia Guo, Niannan Xue, Stefanos Zafeiriou”, IEEE, 2018.

[17] Florian Schroff, Dmitry Kalenichenko, James Philbin, “FaceNet: A Unified Embedding for Face Recognition and Clustering”, IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2015, 2015.

[18] Alexander Hermans, Lucas Beyer, Bastian Leibe, “In Defense of the Triplet Loss for Person Re-Identification”, IEEE, 2017.

[19] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, Network Design, “You Only Look Once: Unified, Real-Time Object Detection”, University of Washington, 2015.

[20] Vaibhaw Singh Chandel, “Selective Search for Object Detection (C++/ Python)”, https://www.learnopencv.com/selective-search-for-object-detection-cpp-python, 2017.