Camera Monitor

This system is built on the Open Computer Vision module – usually referred to as cv2. Install it, after installing python, with the following:

```
pip install opencv-python
```

cv2 is the bit that contains all of the magic.

There is a wide interest in using video cameras for monitoring. There are many commercial systems on the market that do this. Most of those systems have a monthly subscription fee. The fees cover Internet storage of videos and images (usually jpeg or mp4 format) and some form of alerting system. The alerting systems send text messages or an emails when motion occurs in front the one or more cameras and have a system for group storage of images from multiple cameras. The system contained here does all of this and requires no subscription fees. Further, this system is under the control of the user, rather than a commercial entity. Monitoring can be setup on most any computer with an attached video camera.

Recording starts when motion is detected. Recording stops when motion ends. There are options for scaling and tweaking the detection. There are options for group storage on a machine of your choice and the storage machine can be local or one of your choosing on the Internet. Email and text may be sent via gmail – thank you google.

The following is a sample invocation of the system:

```
python3 camera_monitor.py -write-jpgs
```

This will result in jpeg images being written to the current directory when motion occurs in front of the camera. The default is to create mp4 videos and the -write-jpgs option overrides that. There are many options that the user can set to change the behavior of the system. The system provided defaults for options work for most people and need not be set to use the system. Of course, the user has the python code and can modify it as they wish.

gmail and upload options can be added to the command line to have the images sent to a user via gmail or to be uploaded to one of your local computers or to a computer on the Internet. The following is a sample command to setup uploads.

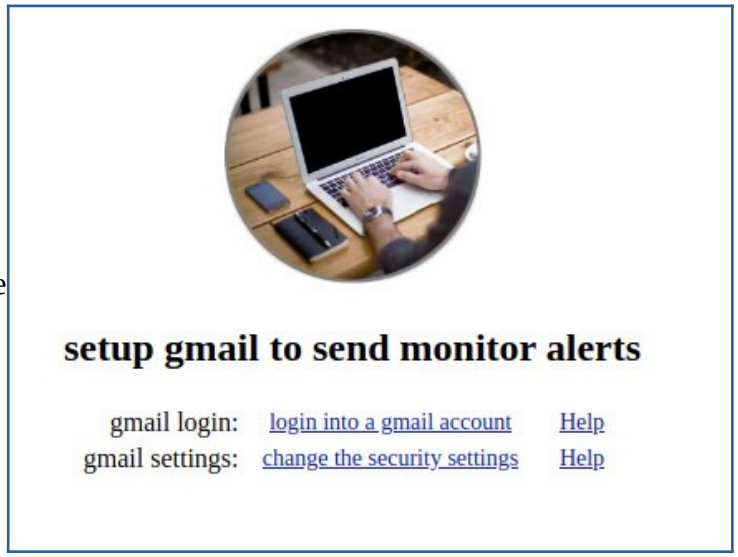
```
python camera_monitor.py -upload-link  http://192.168.1.81:8181/upload.php
```

The systems contains scripts for starting a shared upload server that one might run. We use the php application for this. php is readily available for linux, Apple macs, and windows 10. Apple includes php as part of the free xcode package. On windows one downloads the php server. On linux one uses the apt install features.

The upload.php script is a simple php program that will work under most any http server that supports php. The upload script stores the jpeg or mp4 file in a directory under the name of the camera. The camera name is set with the -camera-name option.

```
python camera_monitor.py -camera-name living-room …
```

One must have a gmail account in order to send the images via email. The option -gmail-browser-config will bring up a local browser window with links for signing in/or creating a gmail account and for setting access by programs to that account. It is best to create a separate account for this.



Keep your gmail account information private and do not leave it exposed on your machines. The camera monitoring program uses encrypted files for storing gmail information. The python3 program “create_gmail_file.py” will create the encrypted file for you. The -gmail-info option uses the encrypted files created by that program.

There is a long list of command line options to allow one to tailor the system to individual needs. Again, the system defaults work for most people. The following command will list the options:

```
python3 camera_monitor.py ?
```

The following is the current list of options:

camera_monitor.py: options

-camera-name camera1
the -camera-name is the name of the directory in which the images will be stored when they are uploaded.

-camera-number 0
the -camera-number is the number of the camera or video
device from which video will be read. In most cases this
number will be 0 (zero) as it is usually the first camera.

-count-between-updates 1
set the count between display screen video updates.
this is used to throttle output on slow machines.

-idle-count 24
set the idle_count for detecting idle conditions.
this is used to determine the end of a video capture.

-jpg-capture-frames 4
often the first frame when motion is detected is blurry.
this sets the capture to be a number of frames after the
motion is detected.

-jpg-min-frames 16
set the minimum number of frames to be read prior to each jpg
file being written. This throttles the number of jpg files.
Depending on the system, there are about 20 frames per second.
Setting this to 1200 would result in one frame per minute.

-min-size-areas 64
set the minimum area used for motion detection. differences
less that this size will be ignored.

-no-display
do not display the video while this program is running.
this is used to improve performance on slow machines.

-one-file
put all of the detected frames in one video file.
video files have the form: motion.00-30-47.1--21-04-2021.mp4.
use ffmpeg to convert to other video formats - webm or mov.
ffmpeg -i motion*.mp4 -f webm m.webm

-reset-count 64
set the number of repeated rectangles to trigger a frame1 reset.
frame1 has to be reset when stationary objects are added or
move into the field of view and subsequently stop.

-scale-factor 1
this number is multiplied by the hardware default frame size.
this is used to reduce the size of the video and jpg files.

-write-addon-count 10
set the write count for frames to be added after motion is detected. the added frames provide continuity between detections.

-write-jpgs
put the detected "motion" frames in jpg files.
jpg files have the form: motion.00-30-47.1--21-04-2021.jpg.
set -min-size-areas to 256 to reduce the number of jpg files.

-upload-link http://yourserver:8181/upload.php
the -upload-link is the server name and webpage to which the images will be uploaded. if you are using a local server the link might be: http://localhost:8181/upload.php
one can use multiple cameras and upload them to one website.

-gmail-browser-config
popup a browser window with two links to gmail. One link is for logging into the gmail account. The other link is for setting the less secure access used for sending email. This access must be enabled to allow email to be sent from this program

-gmail-info gmail.txt
-gmail-info reads the login name and the login password from the named encrypted file. The password for the encrypted file is read from the user (or it can included filename:password). This avoids having the gmail password exposed on the command line and visible with process info. We want to never have the gmail login and login password in the clear/open.

-gmail-recipient must be set as well as -gmail-info if email is to be sent. Run create_gmail_file.py to create an info file.

-gmail-recipient some.one@gmail.com
-gmail-recipient sets the recipient address for mail alerts that are sent when motion is detected. -gmail-info must be set as well as -gmail-recipient if email is to be sent.

Setting Up an Upload Server on Linux

The Apache web server is the server of choice on linux machines. There are many tutorials on setting up Apache. We commonly use a simple php server in local environments. Check if you have php on your system and use “sudo apt install php” if it is not there.

You need to find the IP address of the machine. Use either “ifconfig” or “ip address show”. The IP address is usually something of the form “192.168.1.81”. Most home routers create a pool of local address in the 192.168.1.0/24 address space.

Start the php web server with the following:

```
php -S 192.168.1.81:8181
```

The “:8181” specifies the port number of the server. Web servers have been traditionally set at port 80. It is best to avoid port 80 for this.

Next you have to allow access to this port by other machines in your local group. You have to run a few Unix firewall commands.

```
ufw enable

ufw default allow outgoing

ufw default deny incoming

ufw allow from 192.168.1.0/24 to any port 8181
```

One could modify the last ufw command to only allow specific IP addresses. This is a good option when using an internet based server. On the Internet one has to check on your external IP address.

There is only one file needed to setup the php server as an upload server. The file upload.php has all one needs. It is one page of code and very easy to modify. That php file will move uploaded files into subdirectories in an archive directory. The subdirectories are taken from the camera names specified as options to the program.

I hope this helps. You are on your own – but you already knew that.