Tutorial for Edison Image Tracking

Peidong Chen, Yang Yang, Yitian Hu, Jiayu Guo 12/13/2015

1 Introduction

We can use opencv library to track images using Edison board.

2 Edison initialization

To install opencv2.4 for python on Edison and some other packages needed, you should add AlexT's repo to your intel. To do this, you should configure your Ediosn to fetch packages from the repo by adding listing 2 to /etc/opkg/base-feeds.conf.

Firstly, use vi to open the base-feeds.conf:

Listing 1: bash commands for Edison initialization

```
vi /etc/opkg/base-feeds.conf
```

Then write these following lines at the bottom of the file:

Listing 2: contents in base-feeds.conf

```
src/gz all http://repo.opkg.net/edison/repo/all
src/gz edison http://repo.opkg.net/edison/repo/edison
src/gz core2-32 http://repo.opkg.net/edison/repo/core2-32
```

Then run opkg update command and you should see the below output, which means you're successfully communicating with the repo. Do not run "opkg upgrade", that will overfill your rootfs and is not an intended use case for this repo. Upgrade specific packages with just "opkg install <packagename>" if you want to upgrade something.

Listing 3: bash commands for Edison initialization

```
opkg update
```

The following information should appear:

Listing 4: Update information

```
Downloading http://repo.opkg.net/edison/repo/all/Packages.gz.
Inflating http://repo.opkg.net/edison/repo/all/Packages.gz.
Updated list of available packages in /var/lib/opkg/all.
Downloading http://repo.opkg.net/edison/repo/edison/Packages.gz.
Inflating http://repo.opkg.net/edison/repo/edison/Packages.gz.
Updated list of available packages in /var/lib/opkg/edison.
Downloading http://repo.opkg.net/edison/repo/core2-32/Packages.gz.
Inflating http://repo.opkg.net/edison/repo/core2-32/Packages.gz.
Updated list of available packages in /var/lib/opkg/core2-32.
Downloading http://iotdk.intel.com/repos/1.1/intelgalactic/Packages.
Updated list of available packages in /var/lib/opkg/iotkit.
```

If something went wrong, please to this website: http://alextgalileo.altervista.org/edison-package-repo-configuration-instructions.html

3 Camera preparation

To use camera on intel edison, please get a webcam, which is supported by the uvcvideo module (You can refer to this website:http://www.ideasonboard.org/uvc/#devices). Then on the Arduino expansion board, check that the USB switch on the right side should be flipped up toward the USB Type A (the larger USB connector). Now you can plug the USB webcam to the powered USB port.In

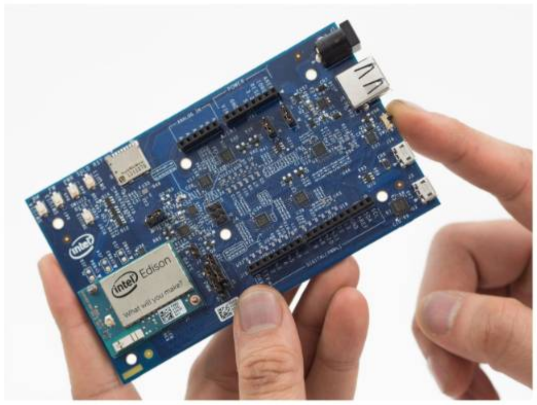


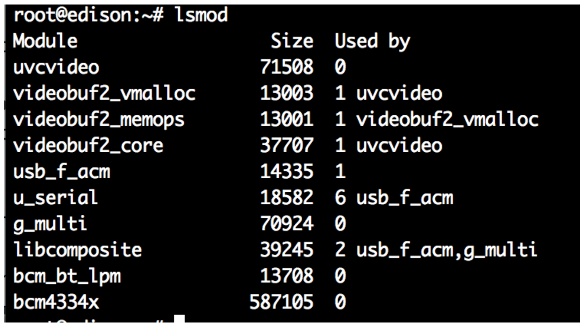
Figure 1: Figure shown of USB switch on arduino expansion board

order to see if the system has loaded the proper modules to manage the webcam, type the following command in the terminal:

Listing 5: Bash command

```
lsmod
```

Some thing like the following picture should appear. The first line shows you camera information and its number is 0. If something went wrong, try check your camer type, update your board with the latest OS image or install uvcvideo kernel module by running "opkg install kernel-module-uvcvideo" on the board.



```
root@edison:~# lsmod
Module                  Size  Used by
uvcvideo               71508  0
videobuf2_vmalloc      13003  1 uvcvideo
videobuf2_memops       13001  1 videobuf2_vmalloc
videobuf2_core         37707  1 uvcvideo
usb_f_acm              14335  1
u_serial               18582  6 usb_f_acm
g_multi                70924  0
libcomposite           39245  2 usb_f_acm,g_multi
bcm_bt_lpm             13708  0
bcm4334x              587105  0
```

Figure 2: bash information for lsmod

4 package preparation

4.1 Install opencv for python on Edison

After add AlexT's repo to our edison, you can install opencv to edison using the command 'opkg install python-numpy opencv python-opencv'. If you want to use opencv in C++, please see the 4.2 for details. We have compared the efficiency of opencv in C++ and python, which is almost the same as the library of opencv for python is also written in python. So we choose to use python in this tutoral.

4.2 Install opencv3.0 C++ library on Edison

Listing 6: bash commands for installing opencv

```
cd ~/PathToOpencv
git clone https://github.com/Itseez/opencv.git
git clone https://github.com/Itseez/opencv_contrib.git
cd opencv
mkdir build
cd build
```

```
cmake -D CMAKE_BUILD_TYPE=Release -D CMAKE_INSTALL_PREFIX=/usr/local -↩
    DOPENCV_EXTRA_MODULES_PATH=/home/root/PathToOpencv/opencv_contrib/↩
    modules  ..
make -j7
cd ~/PathToOpencv/opencv/build/doc/
make -j7 html_docs
cd ~/PathToOpencv/build
make install
```

4.3 Other packages

We need install other packages which are needed for using OpenCV.

```
cd ~
wget https://bootstrap.pypa.io/get-pip.py
python get-pip.py
pip install pyyaml
pip install requests
```

5 Server Build

Our reference image for detecting is store in the server for the purpose of remote control. If you already have a server, you can skip this setup. Otherwise, you can follow the instruction in this step to build a sever on your computer.

For the server on our computer, we need Apache, php, mysql. Here is a post which introduces the way to build server using Mac OSX.
Enable Apache on Mac OS X

```
sudo apachectl start
```

Edit Apache preference

```
cd /etc/apache2/
sudo cp httpd.conf httpd.conf.bak
sudo vim httpd.conf
```

Uncomment the following line (remove #):

```
LoadModule php5_module libexec/apache2/libphp5.so
```

Edit these lines

Listing 7: contents in httpd.conf

```
User Username
Group staff
DocumentRoot "/Users/Username/www/"
<Directory "/Users/Username/www/">
Options Indexes FollowSymLinks Multiviews
MultiviewsMatch Any
AllowOverride All
Require all granted
</Directory>
```

And restart the server to check php compatibility

```
sudo apachectl restart
cd /Users/Username/www
touch phpinfo.php
echo '<?php phpinfo();' > phpinfo.php
```

Install MySQL

```
brew install mysql
```

Connect PHP and MySQL

```
cd /var
sudo mkdir mysql
cd mysql
sudo ln -s /tmp/mysql.sock mysql.sock
```

Setup mysql permission

```
mysql -u root
CREATE USER 'webmaster'@'localhost';
```

Then type in commands in mysql terminal

```
create database edison;
grant all on edison.* to 'webmaster'@'localhost';
```

Setup Edison Server

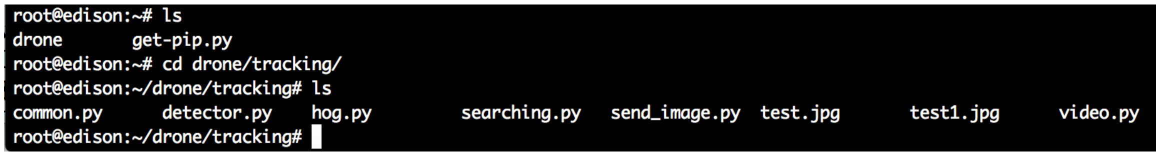
```
cd ~/www
mkdir rest
cd ~/Github/drone
ln -s `pwd`/Server ~/www/rest/api
```

6 Run Image Tracking

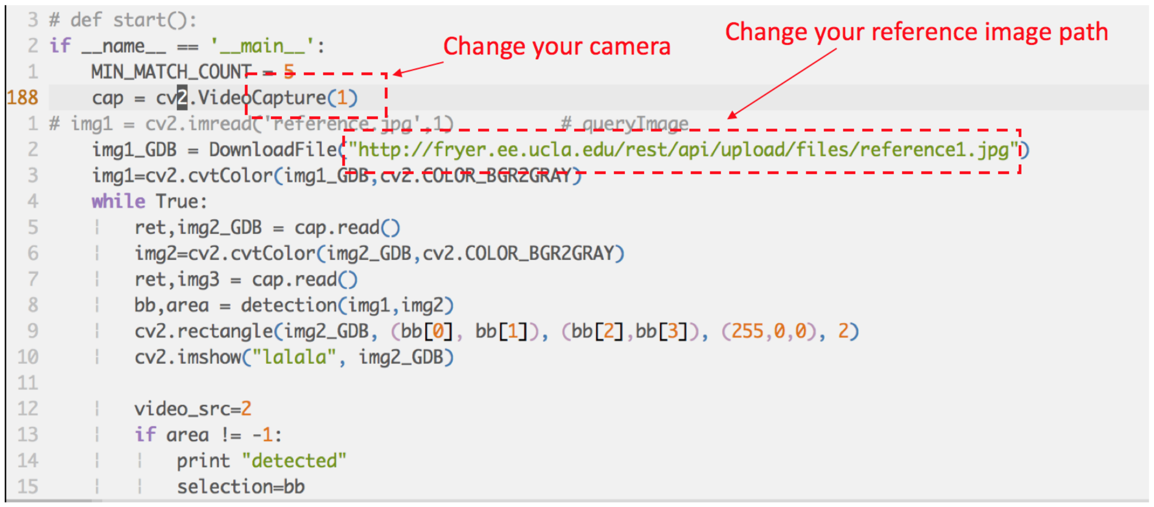
You can download our code by the following command:

```
git clone https://github.com/peidong/drone.git
```

Then you will see our drone folder in current directory. Then use command 'cd' enter the tracking folder.



```
root@edison:~# ls
drone       get-pip.py
root@edison:~# cd drone/tracking/
root@edison:~/drone/tracking# ls
common.py      detector.py    hog.py          searching.py    send_image.py  test.jpg       test1.jpg       video.py
root@edison:~/drone/tracking#
```



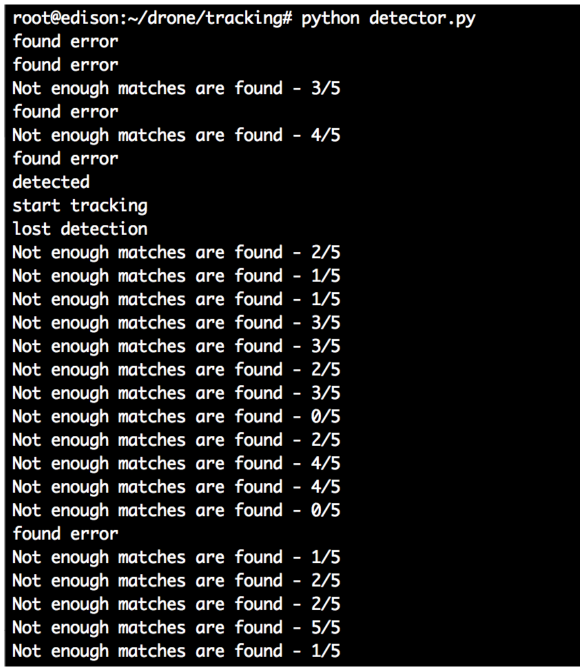
```
# def start():
if __name__ == '__main__':
    MIN_MATCH_COUNT = 5
    cap = cv2.VideoCapture(1)
# img1 = cv2.imread('reference.jpg',1)          # queryImage
    img1_GDB = DownloadFile("http://fryer.ee.ucla.edu/rest/api/upload/files/reference1.jpg")
    img1=cv2.cvtColor(img1_GDB,cv2.COLOR_BGR2GRAY)
    while True:
        ret,img2_GDB = cap.read()
        img2=cv2.cvtColor(img2_GDB,cv2.COLOR_BGR2GRAY)
        ret,img3 = cap.read()
        bb,area = detection(img1,img2)
        cv2.rectangle(img2_GDB, (bb[0], bb[1]), (bb[2],bb[3]), (255,0,0), 2)
        cv2.imshow("lalala", img2_GDB)

        video_src=2
        if area != -1:
            print "detected"
            selection=bb
```

Figure 3: Screen shoot for change image path and camera number

Now you have to change the camera number and reference image path to yours. As shown in the Fig.5, Use any editor you want open ’detector.py’, change the camera number at line 188 and the path to your reference image on server at line 190.

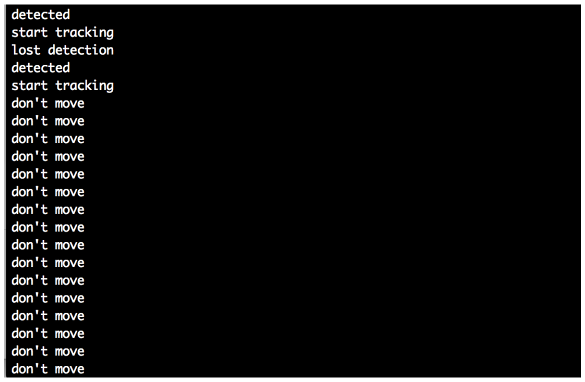
Then you can run detector.py by using command ’python detector.py’.Then you should see it run like this.



```
root@edison:~/drone/tracking# python detector.py
found error
found error
Not enough matches are found - 3/5
found error
Not enough matches are found - 4/5
found error
detected
start tracking
lost detection
Not enough matches are found - 2/5
Not enough matches are found - 1/5
Not enough matches are found - 1/5
Not enough matches are found - 3/5
Not enough matches are found - 3/5
Not enough matches are found - 2/5
Not enough matches are found - 3/5
Not enough matches are found - 0/5
Not enough matches are found - 2/5
Not enough matches are found - 4/5
Not enough matches are found - 4/5
Not enough matches are found - 0/5
found error
Not enough matches are found - 1/5
Not enough matches are found - 2/5
Not enough matches are found - 2/5
Not enough matches are found - 5/5
Not enough matches are found - 1/5
```

Figure 4: Run result

When detecting the reference object from camera, you should see the movement command in terminal.



```
detected
start tracking
lost detection
detected
start tracking
don't move
don't move
don't move
don't move
don't move
don't move
don't move
don't move
don't move
don't move
don't move
don't move
don't move
don't move
don't move
don't move
```

Figure 5: Detection result

7 Some Important Source Code

For algorithm detail you can refer to our project report. We use SURF/SIFT algorith to do object detection. After detecting the object, the location and following frames should be sent to the tracker. We use camshift algorithm for tracking. It will keep tracking the object in the frame and send command according to the movement of the reference object. If the tracker lose the object, it will go back to the detector again.

Here is the implementation of detection algorithm. You can change SIFT in line 2 to SURF to use SURF algorithm, which will be faster.

Listing 8: Detection Algorithm

```
1  def detection(img1, img2):
2      sift = cv2.SIFT()
3      kp1, des1 = sift.detectAndCompute(img1,None)
4      kp2, des2 = sift.detectAndCompute(img2,None)
5      while(des2==None):
6          print "no matches"
7          rst,img2_GDB = cap.read()
8          img2=cv2.cvtColor(img2_GDB,cv2.COLOR_BGR2GRAY)
9          kp2,des2=sift.detectAndCompute(img2,None)
10     # k=cv2.waitKey(0)
11     FLANN_INDEX_KDTREE = 0
12     index_params = dict(algorithm = FLANN_INDEX_KDTREE, trees = 5)
13     search_params = dict(checks = 50)
14     flann = cv2.FlannBasedMatcher(index_params, search_params)
15     matches = flann.knnMatch(des1,des2,k=2)
16     good = []
17     for m,n in matches:
18         if m.distance < 0.7*n.distance:
19             good.append(m)
```

Here is some important code how we implement the tracker. The object detected by the detector are stored in the 'self.selection' in tracker.

Listing 9: Tracker Algorithm

```
if self.selection:
    x0, y0, x1, y1 = self.selection
    self.track_window = (x0, y0, x1-x0, y1-y0)
    hsv_roi = hsv[y0:y1, x0:x1]
    mask_roi = mask[y0:y1, x0:x1]
    hist = cv2.calcHist( [hsv_roi], [0], mask_roi, [16], [0, 180] )
    cv2.normalize(hist, hist, 0, 255, cv2.NORM_MINMAX)
    self.hist = hist.reshape(-1)

    vis_roi = vis[y0:y1, x0:x1]
    cv2.bitwise_not(vis_roi, vis_roi)
    vis[mask == 0] = 0

if self.tracking_state == 1:
    self.selection = None
    prob = cv2.calcBackProject([hsv], [0], self.hist, [0, 180], 1)
    prob &= mask
    term_crit = ( cv2.TERM_CRITERIA_EPS | cv2.TERM_CRITERIA_COUNT, 10, 1 )
    # if prob:
    track_box, self.track_window = cv2.CamShift(prob, self.track_window, term_crit)
```

We print the movement command and store it in our server using codes like the following.

Listing 10: Tracker Algorithm

```
if x_difference < -25:
    print "turn right"
    if formerCommand !=8:
        r = requests.post("http://fryer.ee.ucla.edu/rest/api/control/post/?mac_address=fc:c2:de:3d:7f:af",data = {"manual_control_command":"8"})
        formerCommand = 8
elif x_difference > 25:
    print "turn left"
    if formerCommand !=9:
        r = requests.post("http://fryer.ee.ucla.edu/rest/api/control/post/?mac_address=fc:c2:de:3d:7f:af",data = {"manual_control_command":"9"})
        formerCommand = 9
```