UDC 62.51

N.A.Yesmagambet[1], Yu.N.Antipov[2]

[1] *Ye.A.Buketov Karaganda State University;*
[2] *Technical University of Kaliningrad, Russia*
(E-mail: nurjikyes@gmail.com)

Development recognition system for Unmanned Aerial Vehicle

This article is briefly describe the experimental integration of face recognition system with Unmanned Aerial Vehicle, rest research was covered. Approach considering stream data from uav to ground station over the WiFi network, where raw video data continuously proceed through the object recognition system using OpenCV with Eigenface and Fisherface method.

Key words: UAV, face recognition, opencv, cellular network.

To date, being developed object recognition systems in the field of computer vision. This article is offered the new option of using an object recognition method, in particular a classify human face by the database on unmanned aerial vehicles while operating it through the cellular network.

For complete tasks above we will use our previous experience which briefly described in article «Implementation GStreamer framework with face detection system for Unmanned Aerial Vehicle» [1]. The main updates are: the recognisee system method based on fisherfaces algorithm and how to send data over the cellular network which is much more convenient in such area where UAV can operate widely.

A cellular network or mobile network is a communication network where the last link is wireless. The network is distributed over land areas called cells, each served by at least one fixed-location transceiver, known as a cell site or base station. This base station provides the cell with the network coverage which can be used for transmission of voice, data and others. In a cellular network, each cell uses a different set of frequencies from neighboring cells, to avoid interference and provide guaranteed bandwidth within each cell.

In a cellular radio system, a land area to be supplied with radio service is divided into regular shaped cells, which can be hexagonal, square, circular or some other regular shapes, although hexagonal cells are conventional. Each of these cells is assigned with multiple frequencies (f1 – f6) which have corresponding radio base stations. The group of frequencies can be reused in other cells, provided that the same frequencies are not reused in adjacent neighboring cells as that would cause co-channel interference.

The increased capacity in a cellular network, compared with a network with a single transmitter, comes from the mobile communication switching system developed by Amos Joel of Bell Labs that permitted multiple callers in the same area to use the same frequency by switching calls made using the same frequency to the nearest available cellular tower having that frequency available and from the fact that the same radio frequency can be reused in a different area for a completely different transmission. If there is a single plain transmitter, only one transmission can be used on any given frequency. Unfortunately, there is inevitably some level of interference from the signal from the other cells which use the same frequency. This means that, in a standard FDMA system, there must be at least a one cell gap between cells which reuse the same frequency.

In the simple case of the taxi company, each radio had a manually operated channel selector knob to tune to different frequencies. As the drivers moved around, they would change from channel to channel. The drivers knew which frequency covered approximately what area. When they did not receive a signal from the transmitter, they would try other channels until they found one that worked. The taxi drivers would only speak one at a time, when invited by the base station operator (this is, in a sense, time division multiple access (TDMA)).

When joined together these cells provide radio coverage over a wide geographic area. This enables a large number of portable transceivers (e.g., mobile phones, pagers, etc.) to communicate with each other and with fixed transceivers and telephones anywhere in the network, via base stations, even if some of the transceivers are moving through more than one cell during transmission. The most common example of a cellular network is a mobile phone (cell phone) network. A mobile phone is a portable telephone which receives or makes calls through a cell site (base station), or transmitting tower. Radio waves are used to transfer signals to and from the cell phone.

Modern mobile phone networks use cells because radio frequencies are a limited, shared resource. Cell-sites and handsets change frequency under computer control and use low power transmitters so that the usually limited number of radio frequencies can be simultaneously used by many callers with less interference.

A cellular network is used by the mobile phone operator to achieve both coverage and capacity for their subscribers. Large geographic areas are split into smaller cells to avoid line-of-sight signal loss and to support a large number of active phones in that area. All of the cell sites are connected to telephone exchanges (or switches), which in turn connect to the public telephone network.

In cities, each cell site may have a range of up to approximately 1/2 mile (0.80 km), while in rural areas, the range could be as much as 5 miles (8.0 km). It is possible that in clear open areas, a user may receive signals from a cell site 25 miles (40 km) away.

Since almost all mobile phones use cellular technology, including GSM, CDMA, and AMPS (analog), the term «cell phone» is in some regions, notably the US, used interchangeably with «mobile phone». However, satellite phones are mobile phones that do not communicate directly with a ground-based cellular tower, but may do so indirectly by way of a satellite.

There are a number of different digital cellular technologies, including: Global System for Mobile Communications (GSM), General Packet Radio Service (GPRS), cdmaOne, CDMA2000, Evolution-Data Optimized (EV-DO), Enhanced Data Rates for GSM Evolution (EDGE), Universal Mobile Telecommunications System (UMTS), Digital Enhanced Cordless Telecommunications (DECT), Digital AMPS (IS-136/TDMA), and Integrated Digital Enhanced Network (iDEN). The transition from existing analog to the digital standard followed a very different path in Europe and the US. As a consequence multiple digital standard surfaced in the US, while Europe and many countries converged towards the GSM standard.

For getting it work we need the following hardware and software:

1. Raspberr pi 2 with camera module or any compatible web camera;
2. WiFi Doungle for setting up raspberry pi over the SSH;
3. 3G/4G LTE Modem for access to internet;
4. full operated UAV with all electronics and radio devices;
5. laptop with Linux OS (it's possible to implement it on Windows as well, but for Unix system it's much more easier to install all dependencies and libraries from repositories) for receive video from UAV;
6. properly installed GStreamer framework and opencv libraries on the both side (ground station and RPI).

And we good to go. Next step is start the back side on the RPI.

Face recognition is an easy task for humans. Experiments in [2] have shown, that even one to three day old babies are able to distinguish between known faces. So how hard could it be for a computer? It turns out we know little about human recognition to date. Are inner features (eyes, nose, mouth) or outer features (head shape, hairline) used for a successful face recognition? How do we analyze an image and how does the brain encode it? It was shown by David Hubel and Torsten Wiesel, that our brain has specialized nerve cells responding to specific local features of a scene, such as lines, edges, angles or movement. Since we don't see the world as scattered pieces, our visual cortex must somehow

combine the different sources of information into useful patterns. Automatic face recognition is all about extracting those meaningful features from an image, putting them into a useful representation and performing some kind of classification on them.

Face recognition based on the geometric features [3] of a face is probably the most intuitive approach to face recognition. One of the first automated face recognition systems was described in [4]: marker points (position of eyes, ears, nose, ...) were used to build a feature vector (distance between the points, angle between them, ...). The recognition was performed by calculating the euclidean distance between feature vectors of a probe and reference image. Such a method is robust against changes in illumination by its nature, but has a huge drawback: the accurate registration of the marker points is complicated, even with state of the art algorithms. Some of the latest work on geometric face recognition was carried out. A 22-dimensional feature vector was used and experiments on large datasets have shown, that geometrical features alone my not carry enough information for face recognition.

The Fisherfaces method learns a class-specific transformation matrix, so the they do not capture illumination as obviously as the Eigenfaces method. The Discriminant Analysis instead finds the facial features to discriminate between the persons. It's important to mention, that the performance of the Fisherfaces heavily depends on the input data as well. Practically said: if you learn the Fisherfaces for well-illuminated pictures only and you try to recognize faces in bad-illuminated scenes, then method is likely to find the wrong components (just because those features may not be predominant on bad illuminated images). This is somewhat logical, since the method had no chance to learn the illumination [5].

The Fisherfaces allow a reconstruction of the projected image, just like the Eigenfaces did. But since we only identified the features to distinguish between subjects, you can't expect a nice reconstruction of the original image. For the Fisherfaces method we'll project the sample image onto each of the Fisherfaces instead. So you'll have a nice visualization, which feature each of the Fisherfaces describes:

First we need to learn out system by the given the sample of interesting us object.

```
model = PredictableModel(Fisherfaces(), NearestNeighbor())
vc=cv2.VideoCapture(0)
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_alt_tree.xml')
pathdir='prove/'

number = int(raw_input('How many people in front of camera? \n number:'))
for i in range(number):
nome = raw_input('Hi user'+str(i+1)+' What\'s your name?\n name:')
if not os.path.exists(pathdir+nome): os.makedirs(pathdir+nome)
print ( 'Are you ready for photoset?\n')
print ( 'press the button "S" when you faced in center')
while (1):
ret,frame = vc.read()

gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray, 1.2, 3)
for (x,y,w,h) in faces:
cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)
cv2.imshow('Recognition',frame)


if cv2.waitKey(10) == ord('s'):
break
cv2.destroyAllWindows()


start = time.time()
count = 0
while (True):
ret,frame = vc.read()
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray, 1.2, 3)
for (x,y,w,h) in faces:
```

```
cv2.putText(frame,'Trained', (x,y), cv2.FONT_HERSHEY_SIMPLEX,1,(0,0,250),3,1)
count +=1
cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)
resized_image = cv2.resize(frame[y:y+h,x:x+w], (100, 100))
if count%5 == 0:
print  (pathdir+nome+str(time.time()-start)+'.jpg')
cv2.imwrite( pathdir+nome+'/'+str(time.time()-start)+'.jpg', resized_image );
cv2.imshow('Recognition',frame)
if cv2.waitKey(1) & 0xFF == ord('q'):
break
cv2.destroyAllWindows()
vc.release()
```

In end of this programm you will get an array of data set of images which will be used for recognizing process. The following code below will execute module with the nearest neighbor method and support vector machines classifier with prediction based on expected value and variance [6].

```
model = PredictableModel(Fisherfaces(), NearestNeighbor())
vc=cv2.VideoCapture(0)
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_alt_tree.xml')
def read_images(path, sz=(256,256)):
c = 0
X,y = [], []
folder_names = []
for dirname, dirnames, filenames in os.walk(path):
for subdirname in dirnames:
folder_names.append(subdirname)
subject_path = os.path.join(dirname, subdirname)
for filename in os.listdir(subject_path):
try:
im = cv2.imread(os.path.join(subject_path, filename),
cv2.IMREAD_GRAYSCALE)
# resize to given size (if given)
if (sz is not None):
im = cv2.resize(im, sz)
X.append(np.asarray(im, dtype=np.uint8))
y.append(c)
except IOError, (errno, strerror):
print "I/O error({0}): {1}".format(errno, strerror)
except:
print "Unexpected error:", sys.exc_info()[0]
raise
c = c+1
return [X,y,folder_names]
pathdir='prove/'
[X,y,subject_names] = read_images(pathdir)
list_of_labels = list(xrange(max(y)+1))
subject_dictionary = dict(zip(list_of_labels, subject_names))
model.compute(X,y)
while (1):
rval, frame = vc.read()
img = frame
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray, 1.1, 5)
for (x,y,w,h) in faces:
cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
sampleImage = gray[y:y+h, x:x+w]
sampleImage = cv2.resize(sampleImage, (256,256))
[ predicted_label, generic_classifier_output] = model.predict(sampleImage)
print [ predicted_label, generic_classifier_output]
if int(generic_classifier_output['distances']) <=  700:
                    cv2.putText(img,str(subject_dictionary[predicted_label]), (x,y),
cv2.FONT_HERSHEY_SIMPLEX,1,(0,0,250),3,1)
cv2.imshow('result',img)
if cv2.waitKey(10) == 27:
break
cv2.destroyAllWindows()
vc.release()
```

The last step is starting the streame for sending video over the cellular network, for this we need to start pipeline from the raspberry pi side [7].

```
raspivid -n -w 640 -h 480 -t 0 -o - | gst-launch-1.0 -v fdsrc !
h264parse ! rtph264pay config-interval=10 pt=96 !
 udpsink host=RPI_IP_ADDRESS port=9000
```

And client side on the ground station, start receiving the data

```
gst-launch-1.0 -v udpsrc port=9000 caps='application/x-rtp, media=(string)video,
 clock-rate=(int)90000, encoding-name=(string)H264' ! rtph264depay ! video/x-h264,
 width=640,height=480,framerate=30/1 ! h264parse ! avdec_h264 ! videoconvert !
 autovideosink sync=false
```

Summary. During research and experemential UAV prototyping achieved the following goals:
1. UAV prototype.
2. Recognition system based on Fisherface method.
3. Get streaming video over network from UAV to GSC.

References

1 *Yesmagambet N.* Implementation GStreamer framework with face detection system for Unmanned Aerial Vehicle // Bull. of Karaganda University. Ser. Mathematics. — 2016. — No. 2(82).

2 *Viola P., Jones M.J.* Robust real-time face detection // International Journal of Computer Vision. — 2004. — Vol. 57. — No. 2. — P. 137–154.

3 *Kanade T.* Picture processing system by computer complex and recognition of human faces: PhD thesis. — Kyoto: Kyoto University, 1973.

4 *Turati C., Macchi C.V., Simion F., Leo I.* Newborns face recognition: Role of inner and outer facial features // Child Development. — 2006. — No. 77 (2). — P. 297–311.

5 *Viola P., Jones M.J.* Rapid Object Detection using a Boosted Cascade of Simple Features // Proceedings IEEE Conf. on Computer Vision and Pattern Recognition. — 2001.

6 *Turk M., Pentland A.* Eigenfaces for recognition // Journal of Cognitive Neuroscience. — 1991. — No. 3. — P. 71–86.

7 GStreamer framework documentation. — [ER]. Access mode: https:GStreamer.freedesktop.org.

Н.А.Есмағамбет, Ю.Н.Антипов

Пилотсыз ұшу аппараты үшін түрді тану жүйесін құру

Мақалада пилотсыз ұшу аппаратымен (ПҰА) бет-пішінін анықтау жүйесінің экспериментті зерттеу жұмысы баяндалған. Бұл әдіс ПҰА-дан жер стансасына Wi-Fi арқылы деректер OpenCV және Eigenface, Fisherface әдістерін қолдану арқылы жүзеге асатынын сипаттайды.

Н.А.Есмагамбет, Ю.Н.Антипов

Разработка системы распознавания для беспилотного летательного аппарата

В статье описана экспериментальная разработка интеграции системы распознавания лиц с беспилотным летательным аппаратом (БПЛА). Данный подход предполагает трансляцию видеопотока с БПЛА на наземную станцию через Wi-Fi, где исходные данные проходят через систему распознавания объектов с использованием OpenCV и методов Eigenface и Fisherface.