Vision

Anshul Pahwa
Dhruv Patel

Experiment(s)

The experiments aimed at exploring vision as modality using the front and rear camera on mobile devices. It involved exploring the following :

1) Face Detection
2) Face Recognition
3) Object Detection

Platform

OS : Maemo

It's a linux based OS and powers mobile devices such as Nokia N900 ,Nokia N810(Nokia internet tablet). It has been developed by Nokia in collaboration with open source projects as the Linux kernel, Debian, GNOME, and many more.

Device: Nokia N900

It's is smartphone made by Nokia. This device uses Texas Instruments OMAP3 microprocessor with ARM Cortex-A8. The system has 256 MB of dedicated high performance RAM (Mobile DDR) paired with access to 768 MB swap space managed by the OS. This provides a total of 1 GB of virtual memory. Hence a powerful mobile device.

Libraries used : OpenCV

OpenCV (Open Source Computer Vision Library) is a library of programming functions , developed by Intel and now supported by Willow Garage , which focuses mainly on real time image processing.

Implementation details

The work was done as follows:

1) Configuring the system for the development for maemo.
2) Exploring face detection, face recognition and object detection on the phone.

Configuring the System (for Ubuntu 10.10 and above):

Install OpenCV on system :

Get latest OpenCV library from http://sourceforge.net/projects/opencvlibrary/files/opencv-unix/ and

Step by step installation process is explained here: http://opencv.willowgarage.com/wiki/InstallGuide

Follow the link http://wiki.maemo.org/Documentation/Maemo_5_Final_SDK_Installation, for setting up the environment for Maemo development. The GUI installer , doesnt work well. Even the installation using scripts also shows some glitches(broken packages). So the best way is to do is to :

1)Install Xephyr (follow the steps as given on the above link).

2) Follow the manaul installation for scratchbox and maemo sdk.
3) Login in scratchbox.
4) Install OpenCV in scratchbox (follow the same steps as given below for phone). Make sure you login in FREMANTLE_ARMEL mode (using sb-conf) before installing it.

Configuring the device(Nokia N900) for the use of OpenCV libraries:

To execute the armel binaries generated in scratchbox on phone requires OpenCV installed on it. So , install all the libraries(armel) as given on the page on the page http://maemo.org/packages/source/view/fremantle_extras-devel_free_source/opencv/2.0.0-4/ , on the phone. For that first download all the libraries and their dependencies on the phone and install them in order such that the dependencies are first installed and then the main library. And among main binaries (libhighgui4 , libhighgui-dev , python-opencv , libcvaux , libcvaux-dev , opencv-doc,libcv-dev, libcv4) follow the in which they are given.

First install libhighui4's all dependencies (and further dependencies of the dependencies)and then install it. And do the same for rest of the binaries.
For installing the .deb (armel packages here) use the command:

```
dpkg –i <name of the .deb >
```

Exploring face detection, face recognition and object detection

Face Detection: Here , a real time camera feed or a static image is used as subject to locate faces. The OpenCV library makes it quite easier to detect faces using Haar Cascade classifier. It involves two steps:

1) Training classifiers for face detection using HaarTraining (also known as the Viola-Jones method). This method can be used to train classifiers not only for faces but also for any object . It requires a few hundred positive images(containing the object that we are interested to locate) and negative images. And it takes nearly 6-7 days to generate classifier xml file. Fortunately , classifier for frontal face detection is already available with OpenCV library.

   Reference for Haar Training :
   http://note.sonots.com/SciSoftware/haartraining.html

2) Next is to use , cvHaarDetectObjects function to detect faces using the classifier.

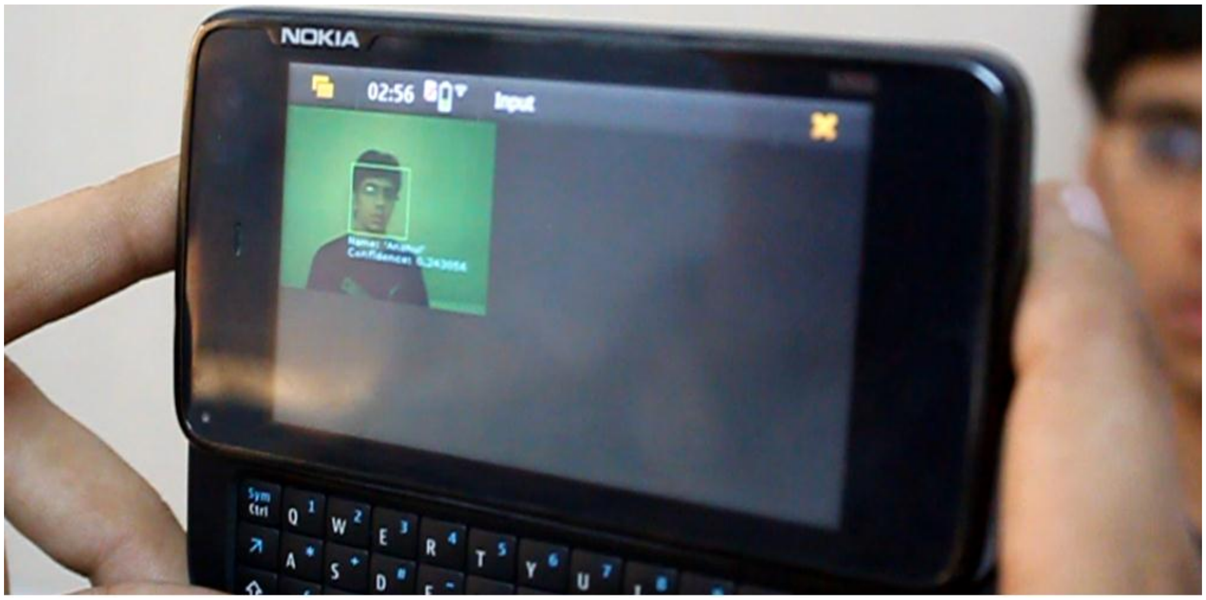
   Reference for using cvHaarDetectObjects:
   http://opencv.willowgarage.com/wiki/FaceDetection

Face Recognition:
In face recognition the detected face is isolated from rest of the image and processed, and then its compared to a database of known faces, to decide who that person is. The most basic technique used for recognizing faces is Eigenfaces or Principal Component Analysis (PCA). The steps that comprise this method are:

1) Compute a "distance" between the new image and each of the example faces.
2) Select the example image that's closest to the new one as the most likely known person.
3) If the distance to that face image is above a threshold, "recognize" the image as that person, otherwise, classify the face as an "unknown" person.

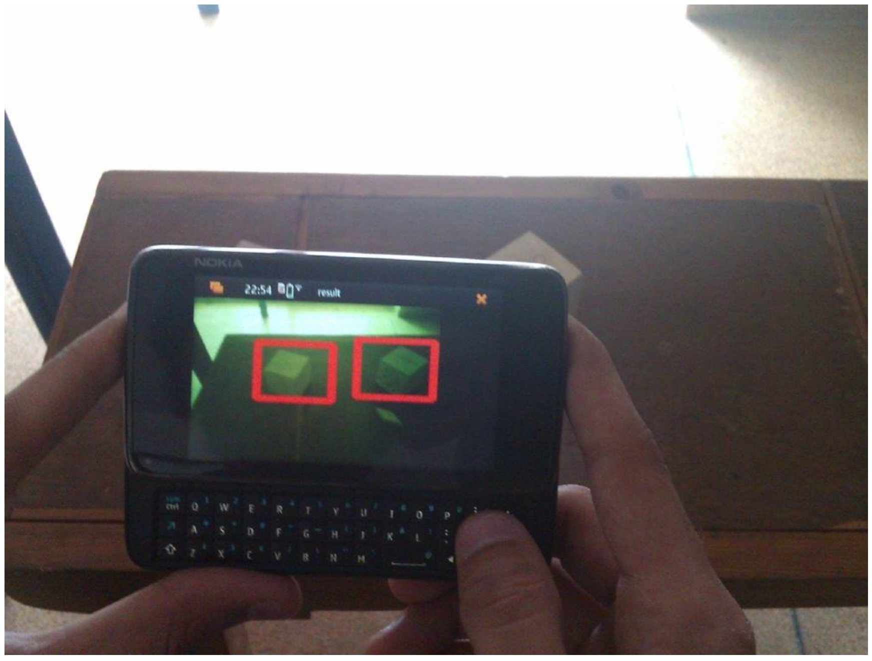
Training for new faces involves:
1) Generating data (xml file) for recognizing the faces.
2) Using this xml file to load image data of the people you want to recognize

Results:
Results for face recognition were 70% accurate.
We tested the program with training for four subjects. In the below two images Manoj And Anshul have been recognized .



Object Detection:

As explained earlier , object detection in OpenCV requires HaarTraining. So we took a cube as a subject for our experiment. In Object detection, we took a few hundreds of positive images of the object (cube in our context) and other negative images and feed them to haartraining which generates an xml file called haar-cascade classifier containing data about different features of object. Then we can use haardetectobjects() function to find the object .



For Object detection, we chose several 4 cm*4 cm wooden cubes to train system and using generated xml cascade, we have been able to recognize the cube via rear-camera of N900 , as you can see it in the above image.

Issues

1) Face Detection: It worked perfectly fine. Make sure that you place the haar-cascade classifier in the same directory as the executable of the face detection program. Using the front camera for face detection, results were not as good as expected, reason being the poor quality of the front camera of the device. Face Recognition: As we used the most basic method for face recognition, It's not much reliable in case of live feed from the camera of the phone, though it works fine in case we supply the static images as input on the phone. Secondly on setting the threshold (for declaring a face unknown requires), the accuracy of the results takes a huge drop. Even the lighting conditions affected the results.
   For getting better results we used pre-processed images that are supplied for training, it helps to neutralize (not completely though) the effect of change in lighting conditions, face rotation and emotion on the face.

2) Cube Detection: The main issue here is the time consumed for the generation of haar-cascade classifier. If we use less number of images, results don't get as accurate as we want and program also classifies some other look-alike objects as object to be recognized. So, we have to train haar-cascade for a week or so with thousand or more positives images of object.

Conclusion

Maemo is a powerful platform. And the freedom that it provides by giving the root access makes it more special. Nokia N900 as a device is good enough (as far as its computational capacity and its rear camera quality is concerned), for it can be used for processor intensive functions of computer vision and image processing.

OpenCV is a great tool for image processing applications on vast range of devices including Nokia N900, given we have dependencies installed on the device. It also provides low-level and performance oriented functions, which are very useful for real-time applications.

Code

To compile the application, source file .cpp and respective haar-cascade file .xml both should be in the same directory.
Now, following command will compile and generate an executable file:

```
gcc -o <executable_filename> <source_filename> `pkg-config opencv --cflags --libs`
```

You can execute the application using following command on terminal:

```
./<executable_filename>
```

If the executable file shall be moved in other directory, cascade .xml file also has to be moved there.

References

Books:

- Open Source Computer Vision Library by Gary R. Bradski, Vadim Pisarevsky, and Jean-Yves Bouguet, Springer, 1st ed. (June, 2006).

Online resources:

- http://www.cs.iit.edu/~agam/cs512/lect-notes/opencv-intro/index.html

For understanding the Eigenface method for face recognition.
http://www.cognotics.com/opencv/servo_2007_series/part_4/index.html