前言：

PCA 是大家经常用来减少数据集的维数，同时保留数据集中对方差贡献最大的特征来达到简化数据集的目的。本文通过使用 PCA 来提取人脸中的特征脸这个例子，来熟悉下在 oepncv 中怎样使用 PCA 这个类。

开发环境：ubuntu12.04+Qt4.8.2+QtCreator2.5.1+opencv2.4.2

PCA 数学理论：

关于 PCA 的理论，资料很多，公式也一大把，本人功底有限，理论方面这里就不列出了。下面主要从应用的角度大概来讲讲具体怎么实现数据集的降维。

1.把原始数据中每个样本用一个向量表示，然后把所有样本组合起来构成一个矩阵。当然了，为了避免样本的单位的影响，样本集需要标准化。

2.求该矩阵的协防差矩阵（关于协方差的介绍可以参考我的博文：一些知识点的初步理解_4(协方差矩阵,ing...)）。

3.求步骤 2 中得到的协方差矩阵的特征值和特征向量。

4.将求出的特征向量按照特征值的大小进行组合形成一个映射矩阵，并根据指定的 PCA 保留的特征个数取出映射矩阵的前 n 行或者前 n 列作为最终的映射矩阵。

5.用步骤 4 的映射矩阵对原始数据进行映射，达到数据降维的目的。

实验说明：

在本次实验实现的过程中，需要用到 opencv 的这些函数，下面简单介绍下这些函数。

Mat Mat::reshape(int cn, int rows=0) const

该函数是改变 Mat 的尺寸，即保持尺寸大小=行数*列数*通道数 不变。其中第一个参数为变换后 Mat 的通道数，如果为 0，代表变换前后通道数不变。第二个参数为变换后 Mat 的行数，如果为 0 也是代表变换前后通道数不变。但是该函数本身不复制数据（这点不是很理解，调用一个 Mat 的 reshape，如果我们不把调用后的 Mat 做为返回值去用，难道此时调用前的 Mat 一点变化都没有？）。

void Mat::convertTo(OutputArray m, int rtype, double alpha=1, double beta=0) const

该函数其实是对原 Mat 的每一个值做一个线性变换。参数 1 为目的矩阵，参数 2 为目 d 矩阵的类型，参数 3 和 4 变换的系数，看完下面的公式就明白了：

PCA::PCA(InputArray data, InputArray mean, int flags, int maxComponents=0)

该构造函数的第一个参数为要进行 PCA 变换的输入 Mat；参数 2 为该 Mat 的均值向量；参数 3 为输入矩阵数据的存储方式，如果其值为 CV_PCA_DATA_AS_ROW 则说明输入 Mat 的每一行代表一个样本，同理当其值为 CV_PCA_DATA_AS_COL 时，代表输入矩阵的每一列为一个样本；最后一个参数为该 PCA 计算时保留的最大主成分的个数。如果是缺省值，则表示所有的成分都保留。

Mat PCA::project(InputArray vec) const

该函数的作用是将输入数据 vec(该数据是用来提取 PCA 特征的原始数据)投影到 PCA 主成分空间中去，返回每一个样本主成分特征组成的矩阵。因为经过 PCA 处理后，原始数据的维数降低了，因此原始数据集中的每一个样本的维数都变了，由改变后的样本集就组成了本函数的返回值。

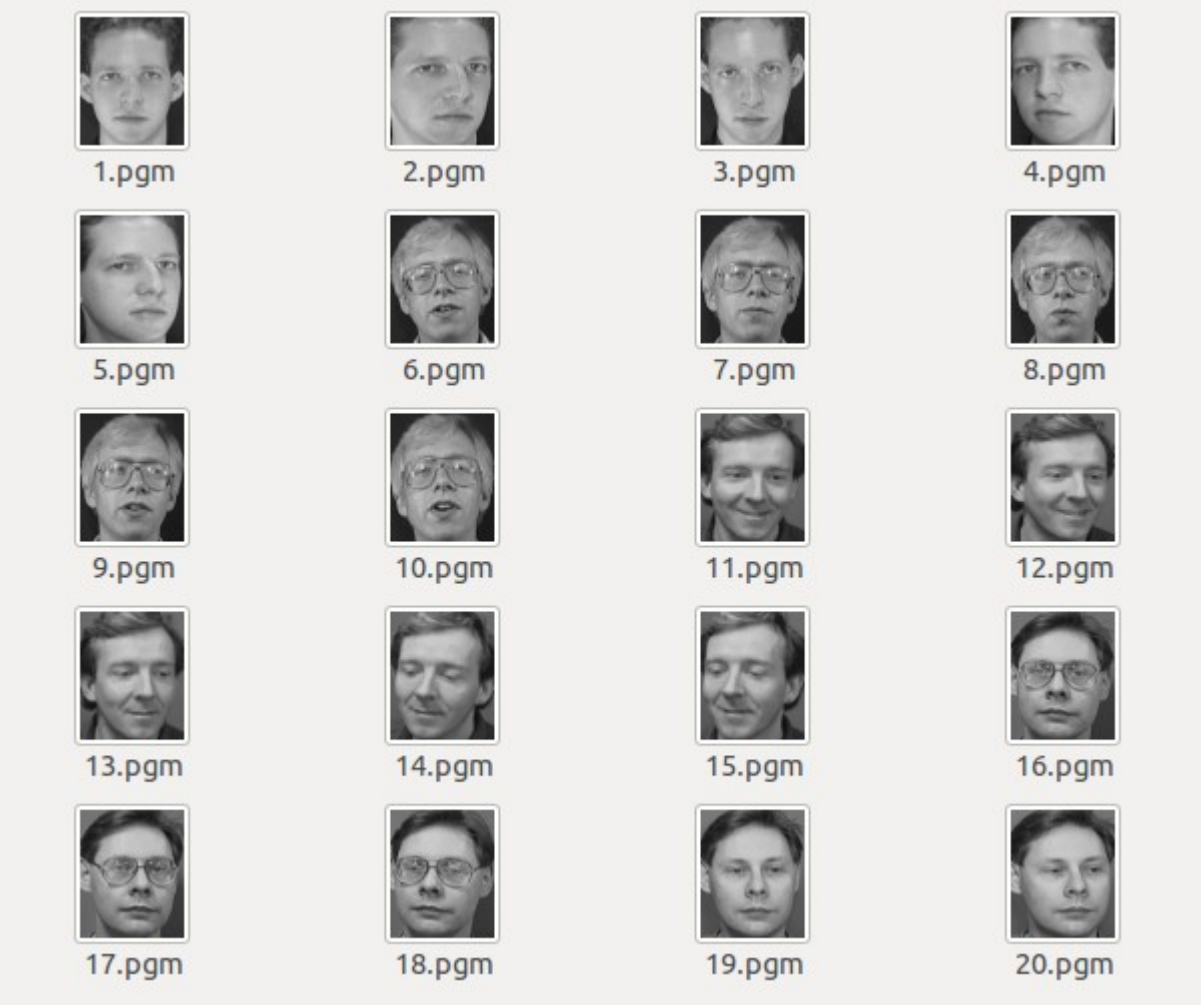
Mat PCA::backProject(InputArray vec) const

一般调用 backProject（ ）函数前需调用 project()函数，因为 backProject()函数的参数 vec 为经过 PCA 投影降维过后的矩阵。 因此 backProject()函数的作用就是用 vec 来重构原始数据集（关于该函数的本质数学实现暂时还不是很了解）。

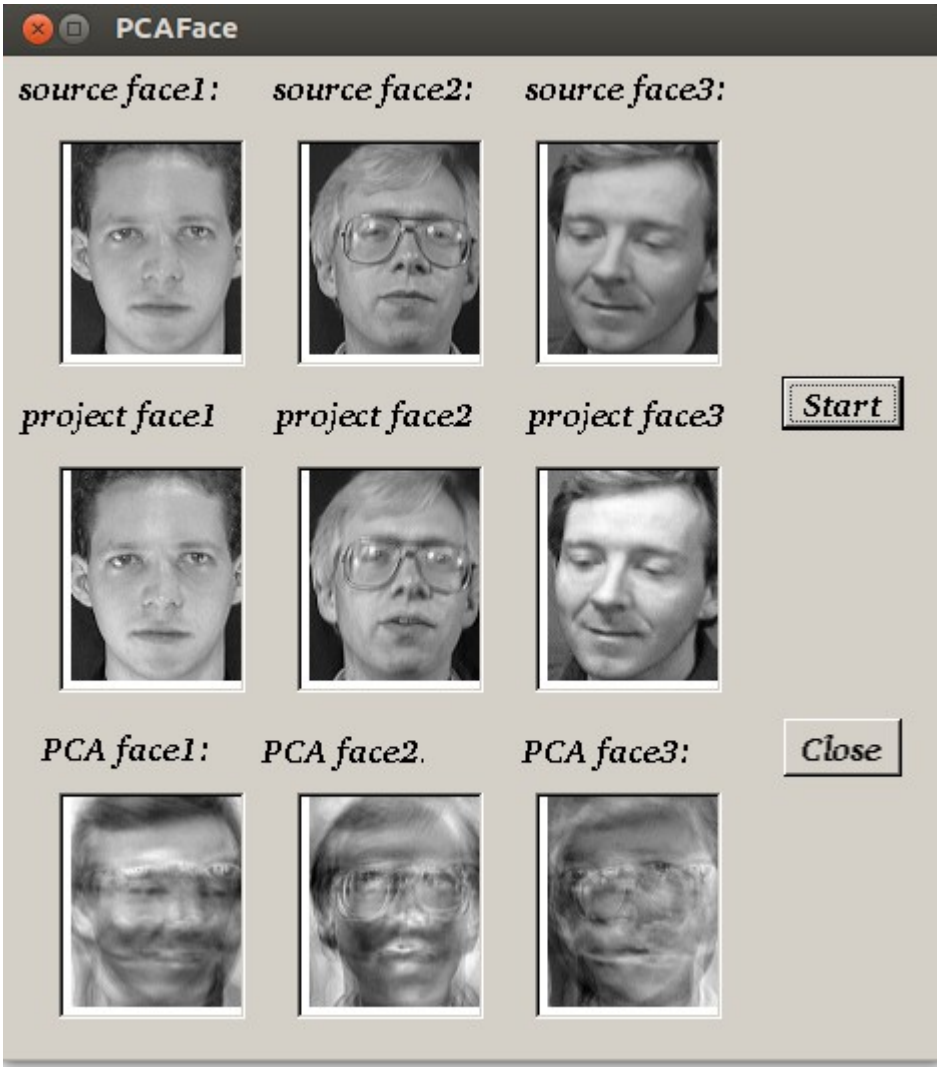
另外 PCA 类中还有几个成员变量，mean,eigenvectors, eigenvalues 等分别对应着原始数据的均值，协方差矩阵的特征值和特征向量。

实验结果：

本次实验是用 4 个人人脸图像，其中每个人分别有 5 张，共计 20 张人脸图片。用这些图片组成原始数据集来提取他们的 PCA 主特征脸。该 20 张图片如下所示：



当运行软件后，单击 start 按钮，该程序的结果显示如下：



其中第一行的 3 张人脸分别为 20 张原图中的 3 张，这里取的是 3 个不同人的。

第二行中显示的 3 张人脸分别为第一行中人脸经过 PCA 投影后，又方向投影过来的人脸图像，仔细观察可以看到第二行的人脸图像整体比第一行的亮度上要亮些，且细节上也有所不同。

第 3 行的人脸图为取的原始数据协方差矩阵特征向量的最前面 3 个，因此这 3 个人脸为最具代表人脸特征的 3 个 PCA 人脸特征。

实验主要部分代码即注释(附录有实验工程 code 下载链接):

pcaface.h:

```
#ifndef PCAFACE_H
#define PCAFACE_H
#include <opencv2/core/core.hpp>
```

```
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/imgproc/imgproc.hpp>

using namespace cv;

#include <QDialog>

namespace Ui {
class PCAFace;
}

class PCAFace : public QDialog
{
    Q_OBJECT

public:
    explicit PCAFace(QWidget *parent = 0);
    ~PCAFace();

    Mat normalize(const Mat& src);


protected:
    void changeEvent(QEvent *e);

private slots:
    void on_startButton_clicked();

    void on_closeButton_clicked();

private:
    Ui::PCAFace *ui;
    Mat src_face1, src_face2, src_face3;
    Mat project_face1, project_face2, project_face3;
    Mat dst;
    Mat pca_face1, pca_face2, pca_face3;
    vector<Mat> src;
    int total;
};
```

```
#endif // PCAFACE_H
```

pcaface.cpp:

```
#include "pcaface.h"
#include "ui_pcaface.h"
#include <QString>
#include <iostream>
#include <stdio.h>

using namespace std;

PCAFace::PCAFace(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::PCAFace)
{
    ui->setupUi(this);
    src_face1 = imread("./images/1.pgm", 0);
    //下面的代码为设置图片显示区域自适应图片的大小
    ui->face1Browser->setFixedHeight(src_face1.rows+1);
    ui->face1Browser->setFixedWidth(src_face1.cols+1);
    ui->face2Browser->setFixedHeight(src_face1.rows+1);
    ui->face2Browser->setFixedWidth(src_face1.cols+1);
    ui->face3Browser->setFixedHeight(src_face1.rows+1);
    ui->face3Browser->setFixedWidth(src_face1.cols+1);

    ui->face4Browser->setFixedHeight(src_face1.rows+1);
    ui->face4Browser->setFixedWidth(src_face1.cols+1);
    ui->face5Browser->setFixedHeight(src_face1.rows+1);
    ui->face5Browser->setFixedWidth(src_face1.cols+1);
    ui->face6Browser->setFixedHeight(src_face1.rows+1);
    ui->face6Browser->setFixedWidth(src_face1.cols+1);

    ui->face7Browser->setFixedHeight(src_face1.rows+1);
```

```
    ui->face7Browser->setFixedWidth(src_face1.cols+1);
    ui->face8Browser->setFixedHeight(src_face1.rows+1);
    ui->face8Browser->setFixedWidth(src_face1.cols+1);
    ui->face9Browser->setFixedHeight(src_face1.rows+1);
    ui->face9Browser->setFixedWidth(src_face1.cols+1);

    for(int i = 1; i <= 15; i++)
    {
        stringstream ss;
        string num;
        ss<<i;//将整数 i 读入字符串流
        ss>>num;//将字符串流中的数据传入 num，这 2 句代码即把数字转换成字符
        string image_name = ("./images/" + num + ".pgm");//需要读取的图片全名
        src.push_back(imread(image_name, 0));
    }
    total= src[0].rows*src[0].cols;
}

PCAFace::~PCAFace()
{
    delete ui;
}

void PCAFace::changeEvent(QEvent *e)
{
    QDialog::changeEvent(e);
    switch (e->type()) {
    case QEvent::LanguageChange:
        ui->retranslateUi(this);
        break;
    default:
        break;
    }
}

//将 Mat 内的内容归一化到 0～255，归一化后的类型为但通道整型
Mat PCAFace::normalize(const Mat& src) {
    Mat srcnorm;
    cv::normalize(src, srcnorm, 0, 255, NORM_MINMAX, CV_8UC1);
```

```
    return srcnorm;
}


void PCAFace::on_startButton_clicked()
{
    //先显示3张原图
    ui->face1Browser->append("<img src=./images/1.pgm>");
    ui->face2Browser->append("<img src=./images/7.pgm>");
    ui->face3Browser->append("<img src=./images/14.pgm>");

    //mat数组用来存放读取进来的所有图片的数据，其中mat的每一列对应1张图片，该实现在下面的for函数中
    Mat mat(total, src.size(), CV_32FC1);
    for(int i = 0; i < src.size(); i++)
    {
        Mat col_tmp = mat.col(i);
        src[i].reshape(1, total).col(0).convertTo(col_tmp, CV_32FC1, 1/255.);
    }
    int number_principal_compent = 12;//保留最大的主成分数
    //构造pca数据结构
    PCA pca(mat, Mat(), CV_PCA_DATA_AS_COL, number_principal_compent);
    //pca.eigenvectors中的每一行代表输入数据协方差矩阵一个特征向量，且是按照该协方差矩阵的特征值进行排序的
    pca_face1 = normalize(pca.eigenvectors.row(0)).reshape(1, src[0].rows);//第一个主成分脸
    imwrite("./result/pca_face1.jpg", pca_face1);//显示主成分特征脸1
    ui->face7Browser->append("<img src=./result/pca_face1.jpg>");

    pca_face2 = normalize(pca.eigenvectors.row(1)).reshape(1, src[0].rows);//第二个主成分脸
    imwrite("./result/pca_face2.jpg", pca_face2);//显示主成分特征脸2
    ui->face8Browser->append("<img src=./result/pca_face2.jpg>");

    pca_face3 = normalize(pca.eigenvectors.row(2)).reshape(1, src[0].rows);//第三个主成分脸
    imwrite("./result/pca_face3.jpg", pca_face3);//显示主成分特征脸3
    ui->face9Browser->append("<img src=./result/pca_face3.jpg>");

    //将原始数据通过PCA方向投影，即通过特征向量的前面几个作用后的数据，因此这里的dst的尺寸变小了
    dst = pca.project(mat);
    //通过方向投影重构原始人脸图像(其本质暂时还没完全弄明白)
```

```
    project_face1 = normalize(pca.backProject(dst).col(0)).reshape(1, src[0].rows);
    imwrite("./result/project_face1.jpg", project_face1);
    ui->face4Browser->append("<img src=./result/project_face1.jpg>");

    project_face2 = normalize(pca.backProject(dst).col(6)).reshape(1, src[0].rows);
    imwrite("./result/project_face2.jpg", project_face2);
    ui->face5Browser->append("<img src=./result/project_face2.jpg>");

    project_face3 = normalize(pca.backProject(dst).col(13)).reshape(1, src[0].rows);
    imwrite("./result/project_face3.jpg", project_face3);
    ui->face6Browser->append("<img src=./result/project_face3.jpg>");
}

void PCAFace::on_closeButton_clicked()
{
    close();
}
```

main.cpp:

```
#include <QApplication>
#include "pcaface.h"

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    PCAFace w;
    w.show();

    return a.exec();
}
```

实验总结：

通过本次实验，对 Opencv 中的 PCA 这个类的使用有了一定的了解。

附录： 实验工程 code 下载。