目录

第 7 章 dlib 实时人脸识别应用开发

7.1 人脸检测

dlib 人脸检测器参考代码：

```
#include <dlib/opencv.h>
#include <dlib/image_processing/frontal_face_detector.h>
#include <dlib/image_processing/render_face_detections.h>
#include <dlib/image_processing.h>
#include <dlib/gui_widgets.h>
#include <opencv/cv.h>
#include <opencv/highgui.h>
#include <opencv2/opencv.hpp>
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/core/core.hpp>
#include <opencv2/imgproc/imgproc.hpp>

using namespace dlib;
using namespace std;
using namespace cv;

int main(int argc,char *argv[])
{
    frontal_face_detector detector = get_frontal_face_detector();
    cv::Mat mimg = cv::imread(argv[1]);
    dlib::cv_image<bgr_pixel> img(mimg);//转成 dlib 格式提取特征
    std::vector<dlib::rectangle> faces = detector(img);
    for (unsigned int i = 0; i < faces.size(); ++i)
        cv::rectangle(mimg, cv::Rect(faces[i].left(), faces[i].top(),
                      faces[i].width(), faces[i].width()), cv::Scalar(0, 0, 255),
                      1, 1, 0);//画矩形框
    cv::imshow("人脸", mimg);
    cv::waitKey(0);
}
```

7.2 提取人脸特征点

dlib 官网有一个训练好的特征模型 shape_predictor_68_face_landmarks.dat，搭建环境时下载的，

人脸特征提取参考代码：

```
#include <dlib/opencv.h>
#include <dlib/image_processing/frontal_face_detector.h>
#include <dlib/image_processing/render_face_detections.h>
#include <dlib/image_processing.h>
#include <dlib/gui_widgets.h>
#include <opencv/cv.h>
#include <opencv/highgui.h>
#include <opencv2/opencv.hpp>
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/core/core.hpp>
#include <opencv2/imgproc/imgproc.hpp>
#include <opencv2/freetype.hpp>
```

```
using namespace dlib;
using namespace std;
using namespace cv;

int main(int argc,char *argv[])
{
    frontal_face_detector detector = get_frontal_face_detector();
    shape_predictor sp;
    //提取特征
    deserialize("shape_predictor_68_face_landmarks.dat") >> sp;
    cv::Mat mimg = cv::imread(argv[1]);
    dlib::cv_image<bgr_pixel> img(mimg);
    std::vector<dlib::rectangle> faces = detector(img);
    std::vector<full_object_detection> shapes;
    for (unsigned long i = 0; i < faces.size(); ++i)
        shapes.push_back(sp(img, faces[i]));

    for (unsigned long j = 0; j < faces.size(); ++j)
    if (!shapes.empty()) {
        for (int i = 0; i < 68; i++)//标记特征
            circle(mimg, cvPoint(shapes[j].part(i).x(),
                   shapes[j].part(i).y()), 1,
                   cv::Scalar(0, 255, 0), -1);
    }
    cv::imshow("Dlib 特征点", mimg);
    waitKey(0);
}
```

如果要绘制出人脸特征，dlib 提供了 render_face_detections(shapes)接口，可以通过 dlib 自带的 image_window 类进行绘制。

7.3 人脸识别

dlib 人脸识别采用了 Resnet 残差神经网络，识别精度高于普通神经网络，同样我们可以到官网去下载训练好的模型 dlib_face_recognition_resnet_model_v1.dat，通过 net()接口返回 128 维人脸特征，然后再通过目标图像也同样得到 128 维人脸特征，将两组特征进行对比即可判断出要识别的对象。

具体实现参考代码：从图像、视频、摄像头识别出目标人物

7.4 dlib 对非物定目标识别之手势识别

7.4.1 编译训练工具

一、找到 imglab 工具源码目录进行配置编译

```
cd tools/imglab/
cmake .
make
```

编译完成后将生成的 imglab 工具拷贝到样本照片所在目录。

二、找到 examples/train_object_detector.cpp 样本训练工具，将其拷贝出来单独进行编译，并生成 train_object_detector 命令。

7.4.1 训练样本采集

一、拍摄样本照片

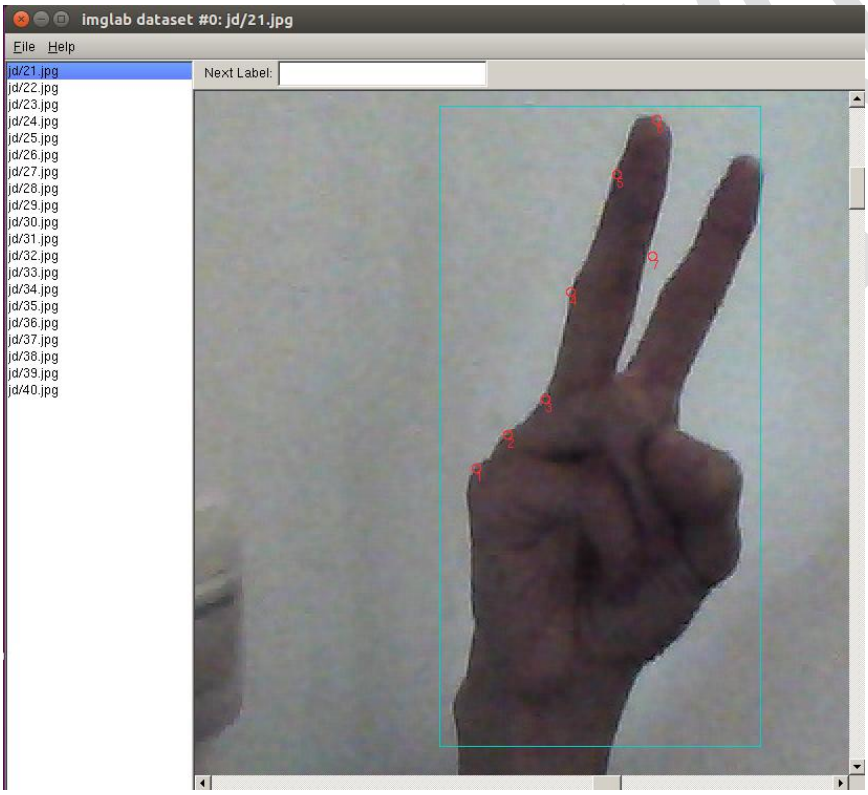
dlib 对样本照片并没有太多要求，会做二次处理，样本拍摄建议在光线充足的情况下完成采集，样本数量越多越好，并将拍好的照片统一放在一个目录，比如 img 下面等待处理。

二、生成 xml 描述文件

```
./imglab -c data.xml img
```

一、手动标记检测目标或特征点

在弹出的图形化工具上面对需要检测的特征进行标记，通过 shift 选择识别对象，双击选中对象以后，右键可以标记特征点，标记完成点 File->save 保存结果到 xml 中。



命令如下：

```
./imglab data.xml
```

如果要标记特征点，需要增加参数

```
./imglab mydataset.xml   --parts   "1 2 3 4 5 6 7 8 9 10"
```

打开 xml 可以查看目标或特征的标记坐标，同时还生成了一个 image_metadata_stylesheet.xsl 样式文件。

二、开始训练样本

```
./train_object_detector -tv data.xml
```

训练结束后会生成 object_detector.svm 模型序列，这个模型就可以用于对象检测了。

三、训练模型测试

一般先从训练样本中随便找张照片进行测试，测试如果能够成功找出目标后，再用于实际的目标识别：

```
./train_object_detector photo.png
```

具体手势识别程序怎么写，请参看代码！