1 C4Feature.hpp

计算图像的C4特征，下面是使用方法直接上demo，就是github project目录下的c4\_feature\_test.cpp

demo:

#include <iostream>

#include <opencv2/core/core.hpp>

#include <opencv2/highgui/highgui.hpp>

#include <opencv2/imgproc/imgproc.hpp>

#include <opencv2/objdetect/objdetect.hpp>

#include <opencv2/ml/ml.hpp>

#include <C4Feature.hpp>

int main() {

cv::Mat src=cv::imread("/Users/wengguifan/Pictures/0428\_nws\_ocr-l-fatal-035.jpg");

C4Feature feature;

float\* histogram=new float[C4Feature::Feature\_size];

feature.Compute(src, histogram);//to generate the feature. The storage is the array of histogram

std::cout << "out of the function" << std::endl;

int pixelNumExpected=24\*18,pixelReal=0;

for(int i=0; i<256; i++) {

pixelReal+=histogram[i];

}

std::cout <<"Pixel num expected:"<<pixelNumExpected<< " Pixel num:"<<pixelReal<< std::endl;

return 0;

}

2 C4Detector.hpp

利用C4扫描整张图像，上demo，就是github project目录下的c4\_detect\_test.cpp

demo:

#include <iostream>

#include <opencv2/core/core.hpp>

#include <opencv2/highgui/highgui.hpp>

#include <opencv2/imgproc/imgproc.hpp>

#include <opencv2/objdetect/objdetect.hpp>

#include <opencv2/ml/ml.hpp>

#include <C4Feature.hpp>

#include <C4Detector.hpp>

int main() {

// cv::Mat src=cv::imread("/Users/wengguifan/Pictures/0428\_nws\_ocr-l-fatal-035.jpg");

cv::Mat src=cv::imread("/Users/wengguifan/pedestrian\_data\_test/pedestrian\_image/INRIA/crop\_000001.png");

C4Feature feature;

//C4Detector(C4Feature \_feature, double \_thresh = 0.8) {

C4Detector detector(feature);

std::vector<cv::Rect>detectedRects;

std::vector<cv::Rect>results;

struct timeval tpstart, tpend;

double timeuse;

gettimeofday(&tpstart, NULL);

detector.MultiDetecte(src,2,0.6,detectedRects);

gettimeofday(&tpend, NULL);

detector.post\_process\_NMS(detectedRects,2, 0.7);

// cout<<"targets\_size:"<<targets.size()<<endl;

detector.post\_process\_NMS(detectedRects,0, 0.7);

detector.post\_process(detectedRects,results,src.rows,src.cols);

timeuse = 1000000 \* (tpend.tv\_sec - tpstart.tv\_sec) + tpend.tv\_usec - tpstart.tv\_usec;

std::cout << "every image:" << timeuse / 1000 << "ms" << std::endl;

std::cout << "detectedRects num:" << detectedRects.size() << std::endl;

for(int i = 0; i < results.size(); i++)

{

cv::rectangle(src, results[i],cv::Scalar(0,255,0),2 );

}

cv::imshow("result",src);

cv::waitKey( 0 );

return 0;

}