**版本信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **版本信息记录表** | | | | |
| 版本号 | 修订内容 | 修订人 | 修订日期 | 审核人 |
| 1.0 | 初版 | 党慧杰 | 2018.08.20 |  |
| 2.0 | 修订 | 党慧杰 | 2018.09.18 |  |
|  |  |  |  |  |

# 实验3—图像比较

## 【实验目的】

1、读取两张图片，对比两张图片的相似度。

## 【实验原理】

**1. OpenCV简介**

OpenCV是一个开放源代码的计算机视觉应用平台，由英特尔公司下属研发中心俄罗斯团队发起该项目，开源BSD证书，OpenCV的目标是实现实时计算机视觉，是一个跨平台的计算机视觉库。其应用领域包括：二维和三维特征工具箱、运动估算、人脸系统识别、姿势识别、人机交互、移动机器人、运动理解、对象鉴别、分割与识别、立体视觉、运动跟踪、增强实现(AR技术)。

**2. 计算颜色直方图**

图像直方图描述的是图像各个灰度级的统计特性，他表示图像每一灰度级与该灰度级出现频率的对应关系，因为灰度级不是连续的，自然，灰度直方图是一个离散函数。横坐标是灰度级g，纵坐标是Ng，如果总的像素是N,灰度级为L，Pg=Ng/N。那么Ng--g构成灰度直方图，Pg--g构成归一化灰度直方图。

函数原型：

void calcHist(const Mat\* images, int nimages, const int\* channels, InputArray mask, OutputArray hist, int dims, const int\* histSize, const float\*\* ranges, bool uniform=true, bool accumulate=false )

参数说明：

* images：输入图像
* nimages：输入图像的个数
* channels：需要统计直方图的第几通道
* mask：掩膜，计算掩膜内的直方图
* hist:输出的直方图数组
* dims：需要统计直方图通道的个数
* histSize：指的是直方图分成多少个区间，就是 bin的个数
* ranges： 统计像素值得区间
* uniform：是否对得到的直方图数组进行归一化处理
* accumulate：在多个图像时，是否累计计算像素值得个数

**3. 归一化**

归一化是一种简化计算的方式，即将有[量纲](https://baike.baidu.com/item/%E9%87%8F%E7%BA%B2/100412" \t "https://baike.baidu.com/item/%E5%BD%92%E4%B8%80%E5%8C%96/_blank)的表达式，经过变换，化为[无量纲](https://baike.baidu.com/item/%E6%97%A0%E9%87%8F%E7%BA%B2/10675963)的表达式，成为[标量](https://baike.baidu.com/item/%E6%A0%87%E9%87%8F/1530843)。

函数原型：

void normalize(const InputArray src, OutputArray dst, double alpha=1, double beta=0, int normType=NORM\_L2, int rtype=-1, InputArray mask=noArray())

参数说明：

* src: 输入数组
* dst: 输出数组，数组的大小和原数组一
* alpha: 用来规范值或规范范围，并且是下限
* beta: 只用来规范范围并且是上限
* normType: 归一化选择的数学公式类型
* rtype: 为负数时，输出数组的type与输入数组的type相同；否则，输出数组与输入数组只是通道数相同，而type=CV\_MAT\_DEPTH(dtype)
* mark: 掩码。选择感兴趣区域，选定后只能对该区域进行操作。

## 【实验环境】

操作系统

Windows 系列 36位/64位

开发环境

Visual Studio 2012、OpenCV

开发语言

C++

## 【实验步骤】

### 一、新建Win32控制台应用程序

1. 点击菜单栏【文件】->【新建】->【项目】或快捷件Ctrl + Shift + N 新建项目，如下图所示：



图 1

2. 在弹出的【新建项目】框中选择模板Visual C++，在中间的栏目中选择【Win32 控制台应用程序】。输入工程名称和保存路径，点击确定。

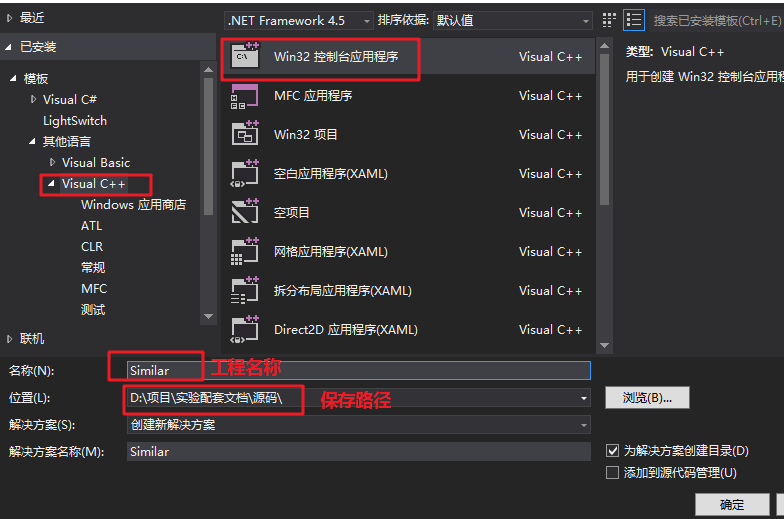


图2

3. 在弹出的【Win32 应用程序向导】对话框中，保持默认配置，直至完成。

### 二、OpenCV配置

在Visual Studio 2012项目中配置OpenCV，参见文档《Opencv的配置》。

### 三、图像比较

**1. 新建ImageHandler类**

1）右击项目，选择【添加】->【类】，如下图所示：

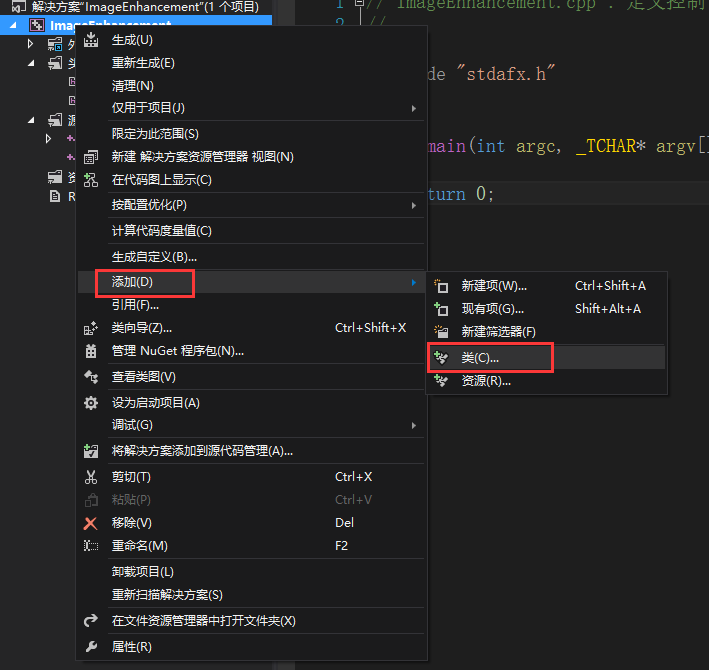


图3

2）选择C++类，点击【添加】

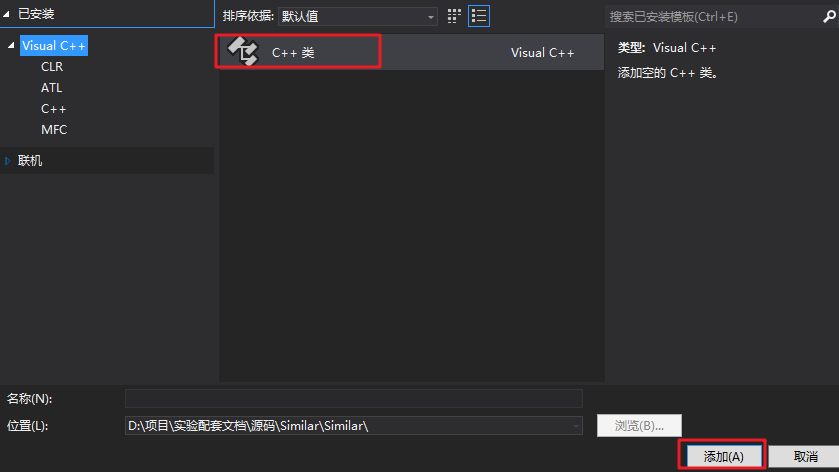


图4

3）输入类名，点击确定，新类就创建完成。

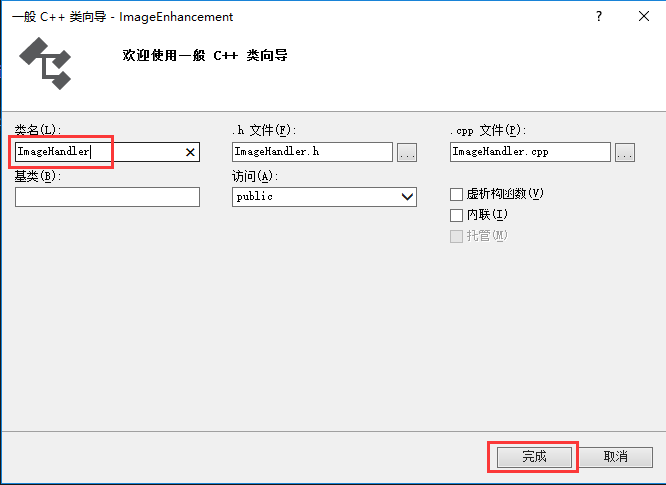


图5

**2. 引入必需的头文件和命名空间**

该方法主要是利用OpenCV提供的函数来完成，因而首先必需添加相关的头文件。打开‘ImageHandler.h’文件。

添加如下头文件：

#include <opencv2/opencv.hpp>

引入命名空间：

using namespace cv;

如下图所示：

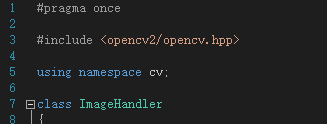


图6

注意：如果不能导入头文件，请检查OpenCV的配置是否正确。

**3. 图像相似度比较的实现**

1）声明函数

打开‘ImageHandler.h’文件，在 ImageHandler 类中添加如下函数声明：

double Similar(Mat frame1, Mat frame2);

其中，参数frame1，frame2为需要比较的两个帧图像。

2）函数实现

算法实现的流程如下：

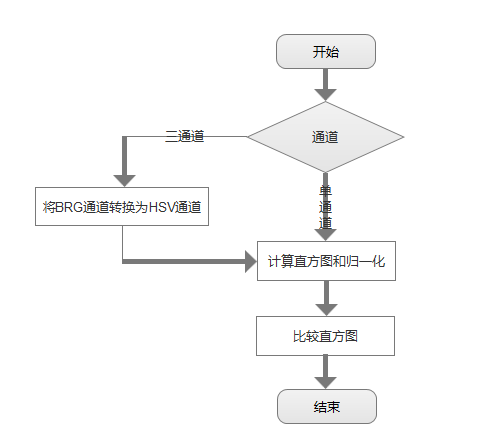


图 7

打开‘ImageHandler.cpp’文件，键入如下程序：

double ImageHandler::Similar(Mat frame1, Mat frame2)

{

if (frame2.channels() == 1) // 单通道

{

int histSize = 256;

float range[] = { 0, 256 };

const float\* histRange = { range };

bool uniform = true;

bool accumulate = false;

cv::Mat hist1, hist2;

// 计算直方图和归一化

cv::calcHist(&frame2, 1, 0, cv::Mat(), hist1, 1, &histSize, &histRange, uniform, accumulate);

cv::normalize(hist1, hist1, 0, 1, cv::NORM\_MINMAX, -1, cv::Mat());

// 计算直方图和归一化

cv::calcHist(&frame1, 1, 0, cv::Mat(), hist2, 1, &histSize, &histRange, uniform, accumulate);

cv::normalize(hist2, hist2, 0, 1, cv::NORM\_MINMAX, -1, cv::Mat());

// 比较直方图

double dSimilarity = cv::compareHist(hist1, hist2, CV\_COMP\_CORREL);

return dSimilarity;

}

else // 三通道

{

// 将BRG通道转换为HSV通道

cv::cvtColor(frame2, frame2, cv::COLOR\_BGR2HSV);

cv::cvtColor(frame1, frame1, cv::COLOR\_BGR2HSV);

int h\_bins = 50, s\_bins = 60;

int histSize[] = { h\_bins, s\_bins };

float h\_ranges[] = { 0, 180 };

float s\_ranges[] = { 0, 256 };

const float\* ranges[] = { h\_ranges, s\_ranges };

int channels[] = { 0, 1 };

cv::MatND hist1, hist2;

// 计算直方图和归一化

cv::calcHist(&frame2, 1, channels, cv::Mat(), hist1, 2, histSize, ranges, true, false);

cv::normalize(hist1, hist1, 0, 1, cv::NORM\_MINMAX, -1, cv::Mat());

// 计算直方图和归一化

cv::calcHist(&frame1, 1, channels, cv::Mat(), hist2, 2, histSize, ranges, true, false);

cv::normalize(hist2, hist2, 0, 1, cv::NORM\_MINMAX, -1, cv::Mat());

// 比较直方图

double dSimilarity = cv::compareHist(hist1, hist2, CV\_COMP\_CORREL);

return dSimilarity;

}

}

3）运行示例

打开‘Similar.cpp’文件，添加头文件‘ImageHandler.h’。在入口函数‘\_tmain’之前，键入如下函数：

void SimilarDemo()

{

Mat img1 = imread("img.jpg");

Mat img2 = imread("img2.jpg");

Mat img3 = imread("img3.jpg");

if (img1.data == NULL || img2.data == NULL) return;

imshow("图1", img1);

imshow("图2", img2);

imshow("图3", img3);

ImageHandler imgHandler;

double result = imgHandler.Similar(img1, img2);

double result2 = imgHandler.Similar(img1, img3);

cout << "图1与图2相似度：" << result << endl;

cout << "图1与图3相似度：" << result2 << endl;

waitKey();

}

在入口函数‘\_tmain’中调用该函数：

int \_tmain(int argc, \_TCHAR\* argv[])

{

SimilarDemo();

return 0;

}

4）运行查看结果

点击vs工具栏中的【本地Windows调试器】（如下图）运行该程序。



图7

程序成功运行后，在控制台将输出图像的比较结果，如下所示：

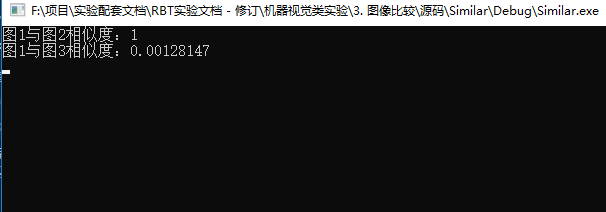


图 8

## 【思考题】

### 选择题

题目1：图像灰度方差说明了图像哪一个属性。（ B ）

A、平均灰度

B、图像对比度

C、图像整体亮度

D、图像细节

题目2：图象与灰度直方图间的对应关系是。（ B ）

A、一一对应

B、多对一

C、一对多

D、都不

### 2、简答题

题目1：什么是图像直方图？

图像直方图描述的是图像各个灰度级的统计特性，他表示图像每一灰度级与该灰度级出现频率的对应关系，因为灰度级不是连续的，自然，灰度直方图是一个离散函数。横坐标是灰度级g，纵坐标是Ng，如果总的像素是N,灰度级为L，Pg=Ng/N。那么Ng--g构成灰度直方图，Pg--g构成归一化灰度直方图。