# 背景模型

该类用于对连续图像的背景进行建模，根据最近frameNum帧图像，对每一像素的灰度值进行统计，使用灰度值出现频率最高的像素值作为背景像素值。

使用方法，首先创建一个Background对象，再使用create方法初始化，使用addframe方法传入新的一帧图像，getBackground方法返回背景图像。

#include "stdafx.h"

#include <string.h>

#include <opencv2/opencv.hpp>

using namespace *std*;

using namespace *cv*;

class Background

{

public:

Background();

~Background();

void create(int \_width, int \_height, int \_frameNum, int \_bins);

void addFrame(*Mat* im);

*Mat* getBackground();

private:

*Mat*\* frames;

int idx;

*Mat* bg;

int\* hist;

int frameNum;

int width;

int height;

int space;

int bins;

bool bInit;

bool bFull;

};

Background::Background()

{

hist = *NULL*;

bInit = false;

}

Background::~Background()

{

if (hist!=*NULL*)

delete[] hist;

if (frames!=*NULL*)

delete[] frames;

}

void Background::create(int \_width, int \_height, int \_frameNum, int \_bins)

{

width = \_width;

height = \_height;

frameNum = \_frameNum;

bins = \_bins;

hist = new int[width\*height\*bins];

*memset*(hist, 0, sizeof(int)\*width\*height\*bins);

frames = new *Mat*[\_frameNum];

idx = 0;

bFull = false;

bInit = true;

}

void Background::addFrame(*Mat* gray)

{

*Mat* im = gray.*clone*();

if (idx >= frameNum)

{

bFull = true;

idx = 0;

}

//将256阶降为bins阶

for (int i = 0; i < im.*rows*;i++)

{

*uchar*\* p = im.*ptr*<*uchar*>(i);

for (int j = 0; j < im.*cols*;j++)

{

p[j] = (int)p[j] / bins;

}

}

//除去frames[idx]的直方图成分，添加当前图像直方图成分

if (!frames[idx].*empty*())

{

for (int i = 0; i < height;i++)

{

*uchar*\* oldP = frames[idx].*ptr*<*uchar*>(i);

*uchar*\* newP = im.*ptr*<*uchar*>(i);

for (int j = 0; j < width;j++)

{

if (oldP[j] != newP[j])

{

hist[i\*width\*bins + j\*bins + oldP[j]] -= 1;

hist[i\*width\*bins + j\*bins + newP[j]] += 1;

}

}

}

}

else

{

for (int i = 0; i < height; i++)

{

*uchar*\* newP = im.*ptr*<*uchar*>(i);

for (int j = 0; j < width; j++)

{

hist[i\*width\*bins + j\*bins + newP[j]] += 1;

}

}

}

frames[idx] = im.*clone*();

idx++;

}

*Mat* Background::getBackground()

{

bg = *Mat*(*Size*(width, height), *CV\_8UC1*, *Scalar*(0));

for (int i = 0; i < height;i++)

{

*uchar*\* p = bg.*ptr*<*uchar*>(i);

for (int j = 0; j < width;j++)

{

int maxIndex = 0;

int maxVal = 0;

for (int k = 0; k < bins;k++)

{

if (maxVal < hist[i\*width\*bins + j\*bins + k])

{

maxIndex = k;

maxVal = hist[i\*width\*bins + j\*bins + k];

}

}

int val = (maxIndex + 1) \* (256 / bins) - 1;

if (val > 255)

{

val = 255;

}

p[j] = (*uchar*)val;

}

}

return bg;

}

#define ROW 135

#define COL 240

#define BIN 16

int *main*()

{

*VideoCapture* cap("F:\\image\_dataset\\video\\handTest.avi");

Background bgModel;

bgModel.create(COL, ROW, 20, BIN);

*Mat* im,gray;

cap >> im;

int idx = 0;

while (im.*data*)

{

*resize*(im, im, *Size*(COL,ROW));

if (idx >= 10)

{

idx = 0;

*cvtColor*(im, gray, *CV\_RGB2GRAY*);

bgModel.addFrame(gray);

*Mat* bg = bgModel.getBackground();

*imshow*("bg", bg);

}

idx++;

*imshow*("im", im);

*waitKey*(10);

cap >> im;

}

return 0;

}