#include<iostream>

#include<highgui\highgui.hpp>

#include<core/core.hpp>

#include<math.h>

using namespace cv;

using namespace std;

// get Gamma transformation look up table

void GetGammaTransLUT(uchar \*pLUT, float Gamma, int iLUTLen)

{

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

{

pLUT[i] = (uchar)(pow((float)i / 255.0, Gamma) \* 255);

}

}

void GammaTrans(uchar \*pSrc, uchar \*pDst, const int iHeight,

const int iWidth, float Gamma)

{

uchar \*pLUT = new uchar[256];

GetGammaTransLUT(pLUT, Gamma, 256);

for (int i = 0; i<iHeight\*iWidth; i++)

{

pDst[i] = (uchar)pLUT[pSrc[i]];

}

delete[]pLUT;

}

int main()

{

Mat image = imread("mars.tif", 0);

Mat image\_Dst = imread("mars.tif", 0);

const int iHeight = image.rows;

const int iWidth = image.cols;

uchar\* pSrc = image.data;//new uchar[iHeight\*iWidth];

uchar\* pDst = image\_Dst.data;//new uchar[iHeight\*iWidth];

GammaTrans(pSrc, pDst, iHeight, iWidth, 0.6);

//namedWindow("Origin",1);

imshow("Origin", image);

//显示图像

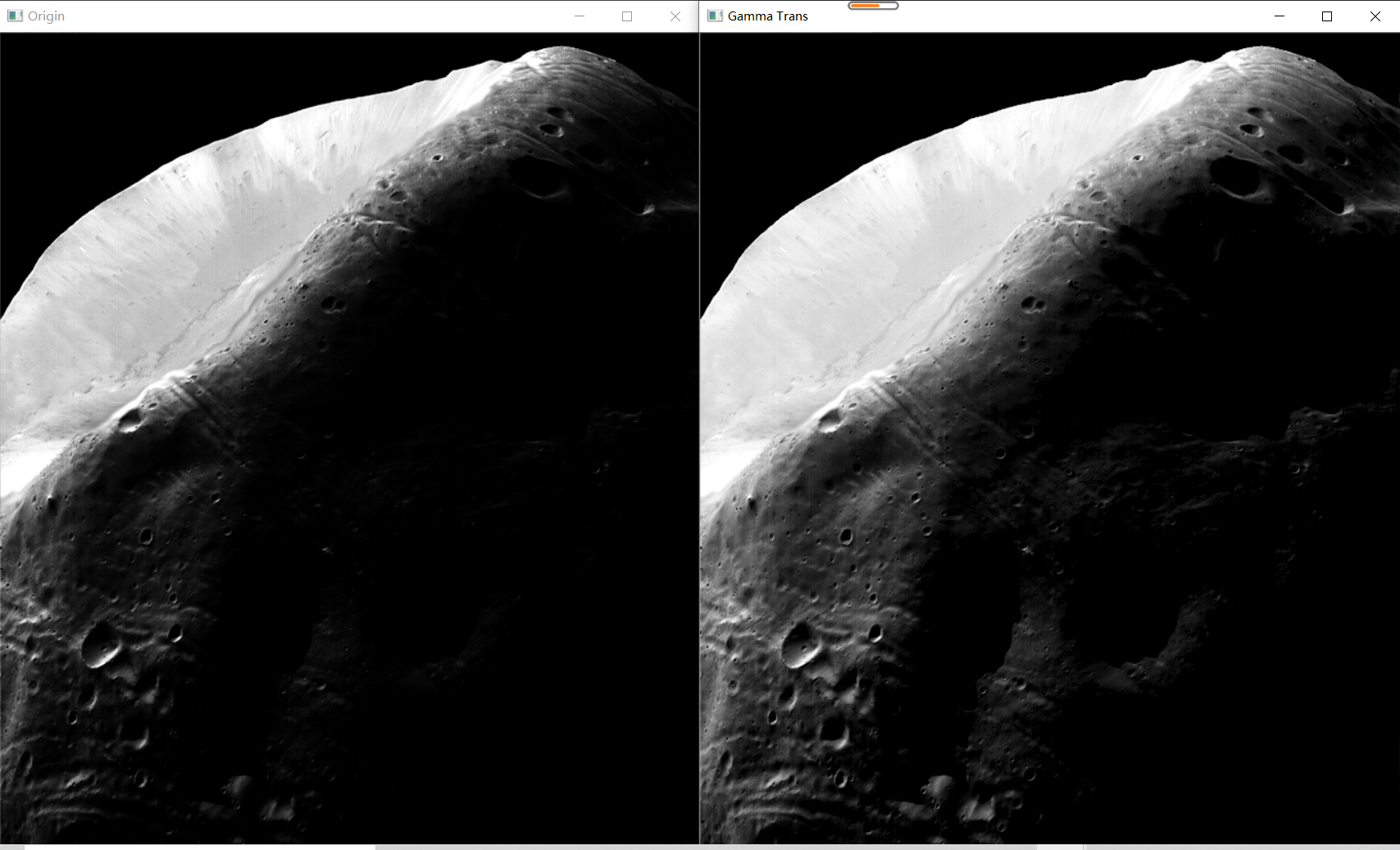
imshow("Gamma Trans", image\_Dst);

//等待按键

waitKey();

return 0;

}



第二题

#include "cv.h"

#include "highgui.h"

#include "cvaux.h"

#include "cxcore.h"

#include "opencv2/opencv.hpp"

//#include "opencv2/imgproc.hpp"

#include <iostream>

#include <string>

using namespace cv;

using namespace std;

//函数名：dividedLinearStrength

//作用：实现分段线性拉伸

//参数：

//matInput：输入图像

//matOutput : 输出图像

//fStart : 分段区间起点

//fEnd ： 分段区间终点

//fSout：映射区间起点

//fEout：映射区间终点

//返回值：无

//注：支持单通道8位灰度图像

void dividedLinearStrength(cv::Mat& matInput, cv::Mat& matOutput, float fStart, float fEnd,

float fSout, float fEout)

{

//计算直线参数

//L1

float fK1 = fSout / fStart;

//L2

float fK2 = (fEout - fSout) / (fEnd - fStart);

float fC2 = fSout - fK2 \* fStart;

//L3

float fK3 = (255.0f - fEout) / (255.0f - fEnd);

float fC3 = 255.0f - fK3 \* 255.0f;

//建立查询表

std::vector<unsigned char> loolUpTable(256);

for (size\_t m = 0; m < 256; m++)

{

if (m < fStart)

{

loolUpTable[m] = static\_cast<unsigned char>(m \* fK1);

}

else if (m > fEnd)

{

loolUpTable[m] = static\_cast<unsigned char>(m \* fK3 + fC3);

}

else

{

loolUpTable[m] = static\_cast<unsigned char>(m \* fK2 + fC2);

}

}

//构造输出图像

matOutput = cv::Mat::zeros(matInput.rows, matInput.cols, matInput.type());

//灰度映射

for (size\_t r = 0; r < matInput.rows; r++)

{

unsigned char\* pInput = matInput.data + r \* matInput.step[0];

unsigned char\* pOutput = matOutput.data + r \* matOutput.step[0];

for (size\_t c = 0; c < matInput.cols; c++)

{

//查表gamma变换

pOutput[c] = loolUpTable[pInput[c]];

}

}

}

int main()

{

cv::Mat matSrc = cv::imread("aerial.tif", cv::IMREAD\_GRAYSCALE);

cv::imshow("原始图", matSrc);

cv::Mat matDLS;

dividedLinearStrength(matSrc, matDLS, 125, 250, 5, 240);

cv::imshow("分段线性变换", matDLS);

cv::waitKey(0);

return 0;

}

