Project5 实验报告

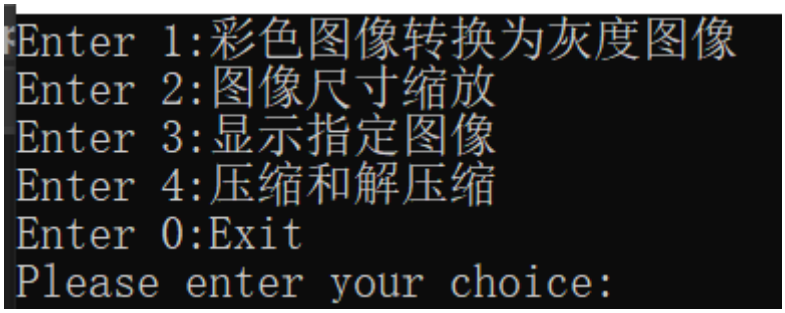
21307035 邓栩瀛

1、程序运行功能简要说明

- 彩色图像转换为灰度图像
- 实现图像尺寸缩放
- 展示指定图像
- 读取并存储图像
- 图像压缩及解压缩

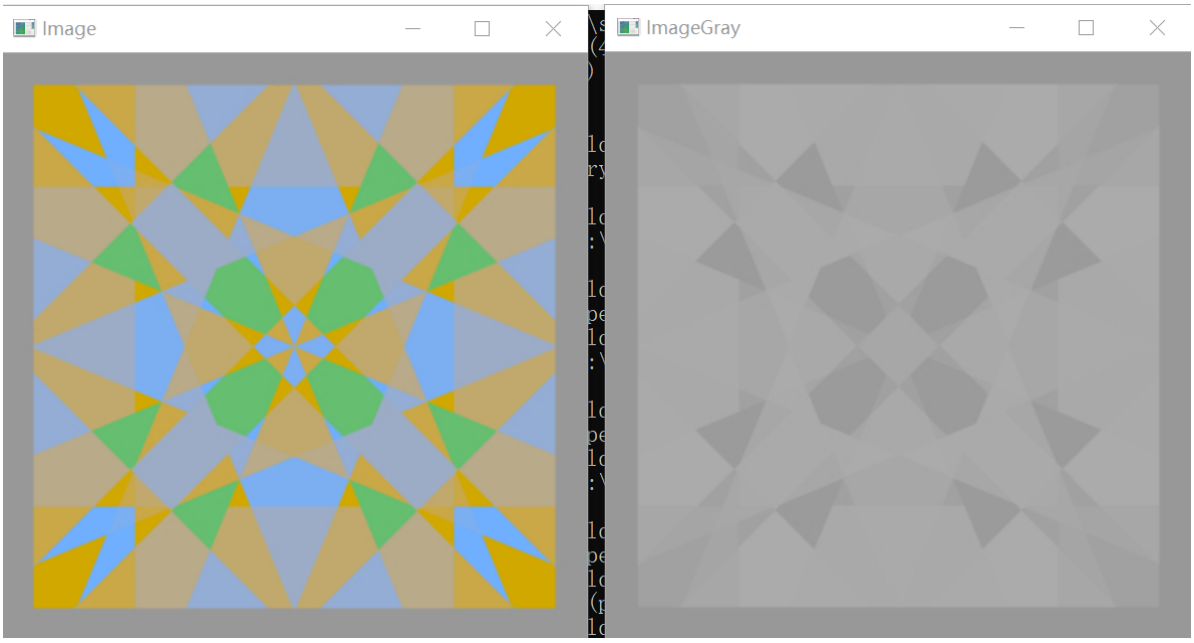
2、程序运行截图，包括计算功能演示、部分实际运行结果展示、命令行或交互式界面效果等。

（1）程序运行截图



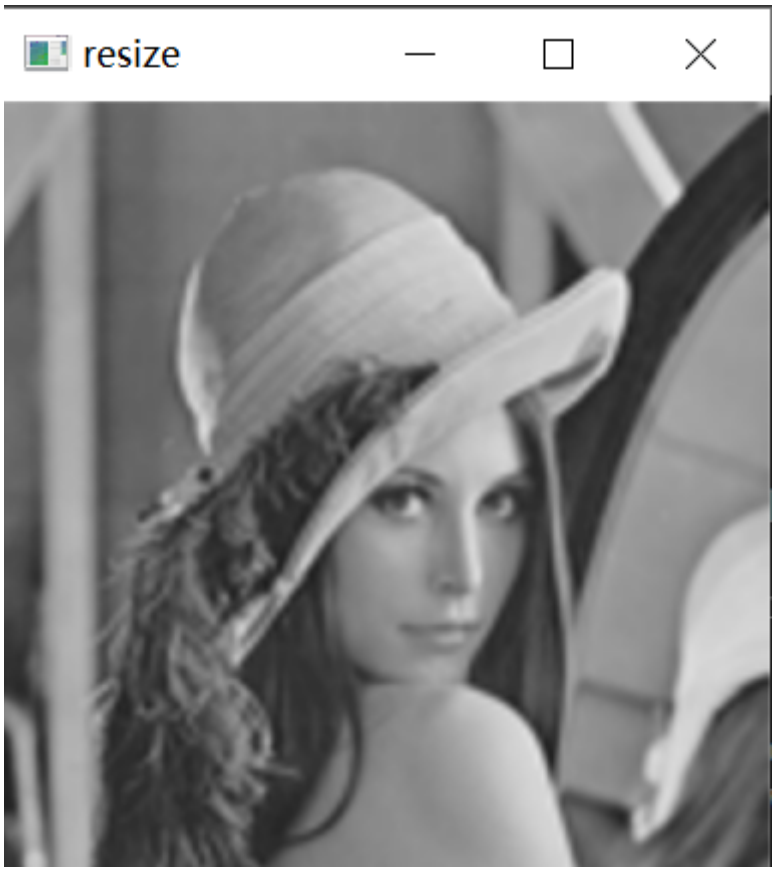
（2）程序功能展示

①彩色图像转换为灰度图像



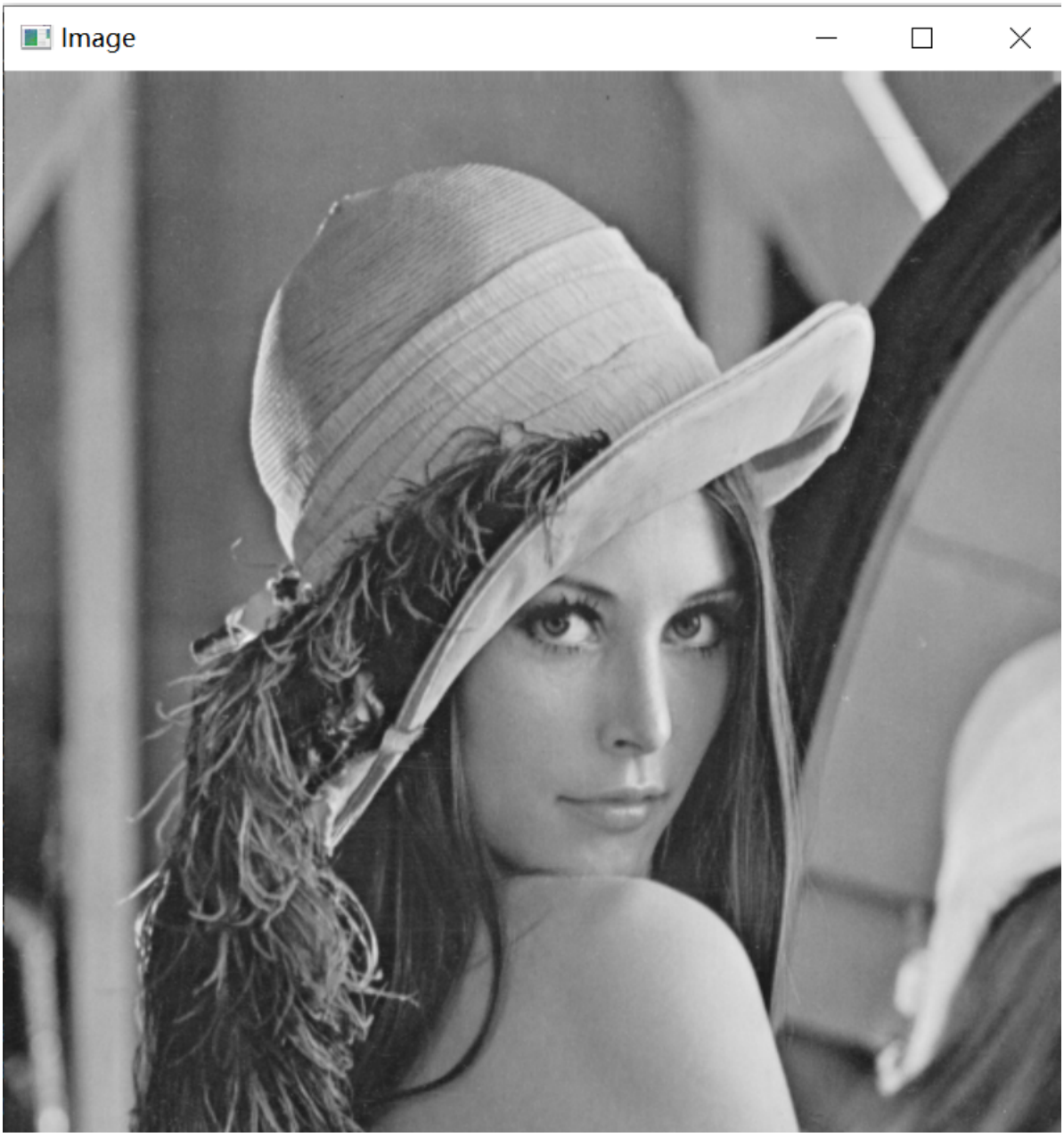
②图像尺寸缩放

```
Please enter your choice:
2
Please enter the filename of the image:
lena-128-gray.ppm
Please enter the length and height to resize:
256 256
Please enter the filename to store:
lena-256-gray.ppm
```

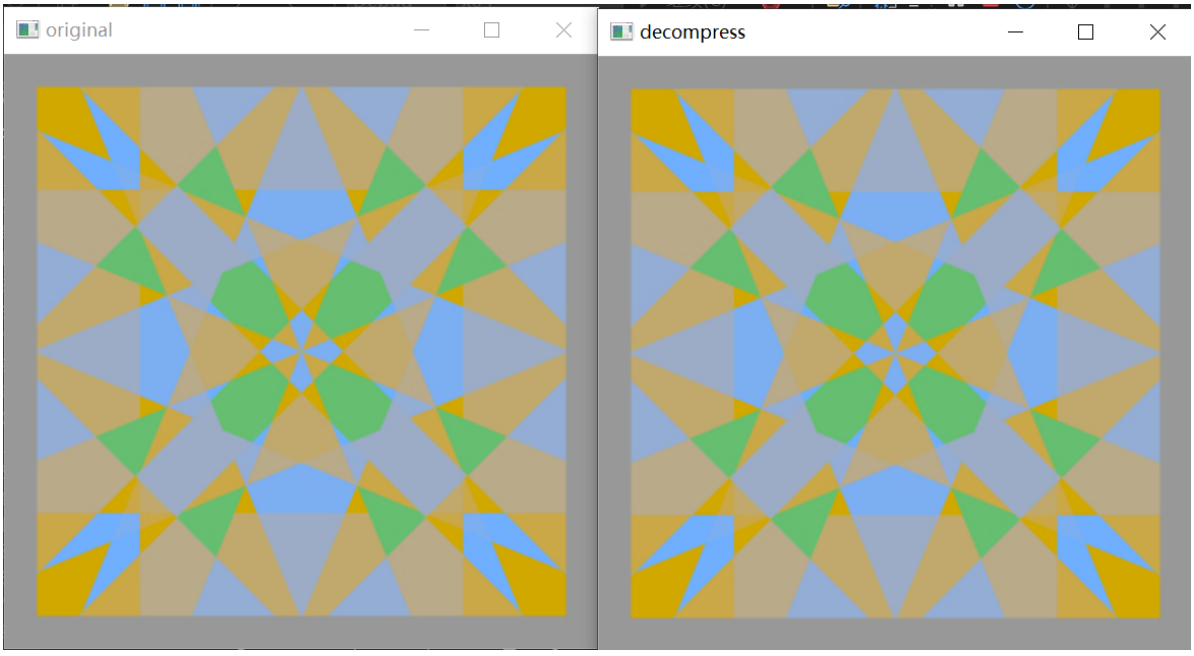


③显示指定图像

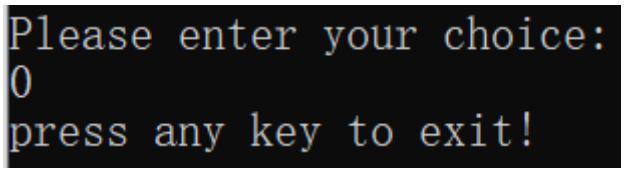
```
Please enter your choice:
3
Please enter the filename of the image:
lena-512-gray.ppm
```



④压缩及解压缩



⑤退出程序



3、部分关键代码及其说明。

(1) 彩色图像转换为灰度图像

```
void ConvertRGB2GRAY()
{
    Mat image, imageGray;
    string path;
    cout << "Please enter the filename of the image:" << endl;
    cin >> path;
    image = imread(path);
    if (!image.data || image.channels() != 3)
    {
        cout << "Error" << endl;
        return;
    }
    //创建一张单通道的灰度图像
    imageGray = Mat::zeros(image.size(), CV_8UC1);
    //取出存储图像像素的数组的指针
    uchar* pointImage = image.data;
    uchar* pointImageGray = imageGray.data;
    //取出图像每行所占的字节数
    size_t stepImage = image.step;
    size_t stepImageGray = imageGray.step;
    for (int i = 0; i < imageGray.rows; i++)
    {
        for (int j = 0; j < imageGray.cols; j++)
        {
            pointImageGray[i * stepImageGray + j] =
                (uchar)(0.114 * pointImage[i * stepImage + 3 * j] +
                    0.587 * pointImage[i * stepImage + 3 * j + 1] +
                    0.299 * pointImage[i * stepImage + 3 * j + 2]);
        }
    }
    string writepath;
    cout << "Please enter the filename to store:" << endl;
    cin >> writepath;
    imwrite(writepath,imageGray);
    imshow("gray", imageGray);
    waitKey(0);
}
```

(2) 图像尺寸缩放

```
Mat my_resize(Mat& img, int width, int height)
{
    // 1, 定义缩放后的图像大小，行列缩放比例
    Mat output = Mat::zeros(Size(width, height), CV_8UC1);
    float width_scale = (float)img.cols / width;      // 列缩放比例
    float height_scale = (float)img.rows / height;    // 行缩放比例

    // 2, 采样
    for (int i = 0; i < height; i++)
    {
        for (int j = 0; j < width; j++)
        {
            output.at<uchar>(i, j) = img.at<uchar>(round(i * height_scale),
round(j * width_scale));
        }
    }
    return output;
}
```

(3) 显示指定图像

```
void display()
{
    cout << "Please enter the filename of the image:" << endl;
    string path;
    cin >> path;
    Mat img = imread(path);
    imshow("Image", img);
    waitKey(0);
}
```

(4) 图像压缩及解压缩

```
void compress_and_decompress()
{
    Mat img;
    cout << "Please enter the filename of the image:" << endl;
    string path;
    cin >> path;
    img = imread(path);
    int quality = 50; //压缩比率0~100
    vector<uint8_t> imageData;
    vector<int> compress_params;
    compress_params.push_back(IMWRITE_PNG_COMPRESSION);
    compress_params.push_back(quality);
    Mat imgup(img.size() * 2, img.type());
    for (int row = 0, rowd = 0; row < img.rows && rowd < img.rows * 2; rowd +=
2, row++)
    {
        for (int col = 0, cold = 0; col < img.cols && cold < img.cols * 2; cold
+= 2, col++)
        {
```

```
            imgup.at<Vec3b>(rowd, cold)[0] = img.at<Vec3b>(row, col)[0];
            imgup.at<Vec3b>(rowd, cold)[1] = img.at<Vec3b>(row, col)[1];
            imgup.at<Vec3b>(rowd, cold)[2] = img.at<Vec3b>(row, col)[2];

            imgup.at<Vec3b>(rowd + 1, cold)[0] = img.at<Vec3b>(row, col)[0];
            imgup.at<Vec3b>(rowd + 1, cold)[1] = img.at<Vec3b>(row, col)[1];
            imgup.at<Vec3b>(rowd + 1, cold)[2] = img.at<Vec3b>(row, col)[2];

            imgup.at<Vec3b>(rowd, cold + 1)[0] = img.at<Vec3b>(row, col)[0];
            imgup.at<Vec3b>(rowd, cold + 1)[1] = img.at<Vec3b>(row, col)[1];
            imgup.at<Vec3b>(rowd, cold + 1)[2] = img.at<Vec3b>(row, col)[2];

            imgup.at<Vec3b>(rowd + 1, cold + 1)[0] = img.at<Vec3b>(row, col)[0];
            imgup.at<Vec3b>(rowd + 1, cold + 1)[1] = img.at<Vec3b>(row, col)[1];
            imgup.at<Vec3b>(rowd + 1, cold + 1)[2] = img.at<Vec3b>(row, col)[2];
        }
    }
    Mat imgdown(img.size() / 2, img.type());
    for (int row = 0, rowd = 0; row < img.rows && rowd < img.rows / 2; row += 2,
rowd++)
    {
        for (int col = 0, cold = 0; col < img.cols && cold < img.cols / 2; col
+= 2, cold++)
        {
            imgdown.at<Vec3b>(rowd, cold)[0] = img.at<Vec3b>(row, col)[0];
            imgdown.at<Vec3b>(rowd, cold)[1] = img.at<Vec3b>(row, col)[1];
            imgdown.at<Vec3b>(rowd, cold)[2] = img.at<Vec3b>(row, col)[2];
        }
    }
    imencode(".png", img, imageData, compress_params);
    vector<uint8_t> p_data = imageData;
    Mat image = imdecode(p_data, -1);
    imshow("original", img);
    imshow("decompress", image);
    waitKey(0);
}
```

4、程序运行方式简要说明。

通过该程序实现图像的读入、存储及输出，图像的压缩及解压缩，并可实现将彩色图像转换为灰度图像以及更改图像的尺寸。