SeamCarving 项目报告

计53 王润基 2015011279

实现功能

- 图像缩小
- 双向放大
- 对象移除和保留
- 三种算子：差分，Sobel，Laplace

效果

见`results`文件夹

遇到的问题

曾经遇到过，图片一直有明显的条纹。原因是算法找到的seam中经常包括相同的部分。如图。

经过痛苦的调试，原因出在算子上。之前使用OpenCV的filter2d函数输出的是Mat1b，即每个像素是一个byte（CV_8U），只有256种取值，这导致能量区分度太小。之后改为每个像素2byte（CV_16S），有65536种取值。就消除了这个Bug。

代码段

- DP

```
const int MAXN = 4000;
int f[MAXN][MAXN];

void dp(cv::Mat1i const& imat) {
    int m = imat.size[0];
    int n = imat.size[1];
    assert(m + 1 < MAXN && n + 1 < MAXN);
    for(int j=0; j<n; ++j)
        f[0][j] = imat.at<int>(0, j);
    for(int i=1; i<m; ++i)
        for(int j=0; j<n; ++j)
        {
            int &ff = f[i][j];
            const int* begin = &f[i-1][std::max(0, j - 1)];
            const int* end   = &f[i-1][std::min(n, j + 2)];
            ff = *std::min_element(begin, end);
            ff += imat.at<int>(i, j);
        }
}
```

- 对象移除

  使用另一个Mask图片输入，作用在能量图上。保留区域置为-inf，移除区域置为inf。

```
cv::Mat1i applyMask (cv::Mat1i const& energy, cv::Mat1b const& mask)
{
    assert(energy.size == mask.size);
    cv::Mat1i rst = cv::Mat::zeros(mask.size[0], mask.size[1], CV_32S);
    rst.forEach([&](int& pixel, const int *pos){
        if(mask.at<uchar>(pos) == KEEP)
            pixel = inf;
        else if(mask.at<uchar>(pos) == DELETE)
            pixel = -inf;
        else
            pixel = energy.at<int>(pos);
    });
    return rst;
}
```

- 找到最小的k个不相交seam

  DP，回溯，找到一条seam，在能量图上将这个seam及其左右1格均置为inf。重复以上操作。

```
Seams findMinColSeams (cv::Mat1i const& imat0, size_t k)
{
    cv::Mat1i imat = imat0.clone();
    int m = imat.size[0];
    int n = imat.size[1];

    Seams seams;
    seams.reserve(k);
    while(k--)
    {
        dp(imat);
        vector<int> cols((size_t)m);
        cols[m-1] = (int)(std::min_element(f[m-1], f[m-1] + n) - f[m-
1]);
        auto fmin = f[m-1][cols[m-1]];
        if(fmin >= inf)
            break;
        cerr << fmin << endl;
        for(int i=m-2; i>=0; --i) {
            int j = cols[i + 1];
            const int* begin = &f[i][std::max(0, j-1)];
            const int* end   = &f[i][std::min(n, j+2)];
            cols[i] = int(std::min_element(begin, end) - f[i]);
        }
        for(int i=0; i<m; ++i) {
            int const& col = cols[i];
            imat.at<int>(i, col) = inf;
            imat.at<int>(i, std::max(0, col - 1)) = inf;
            imat.at<int>(i, std::min(n-1, col + 1)) = inf;
        }
        seams.push_back(cols);
    }
    return seams;
}
```

- 放大一次

  找到前seamSize小的seam，分别插入一列。

```
cv::Mat seamCarvingEnlargeCol (cv::Mat const& image, size_t seamSize,
bool trans = false)
{
    if(seamSize == 0)
        return image;
    auto imat = makeIMat(image);
    auto seams = findMinColSeams(imat, seamSize);
    auto mask = cv::Mat::zeros(image.size[0], image.size[1], CV_8U);
    if(show)
        printSeams(image, seams, mask, trans, "image");
    return addSeams(image, seams);
}
```