第一次大作业实验报告

实现的功能

1. 使用计图实现卷积；
2. 自主完成建图，但可使用开源库如 maxflow 完成 GraphCut；
3. 自主构建泊松融合矩阵、通过开源算法进行矩阵求解。

具体实现

1. 位置匹配。

我们需要提取待补全图像距离缺失区域在 K 个像素内的区域 B，通过 opencv dilate 函数来实现。论文中这个数取了 80, 我们在 opencv dilate 中取核的大小为 60*60, interation=1。还尝试了另外一种核，取半径为 1 的核，iteration=40 次，效果和之前接近。
采用 jittor 的元算子计算误差，然后取误差最小的区域。

2. 计算融合边界

在上一步得到了候选区域后，采用 graph cut 算法得到融合边界。边权重为重叠部分的差，最大流用 pymaxflow 计算得到，进而得到最小割，即为融合边界。

3. 自然融合

采用泊松融合算法计算。我们先构造稀疏矩阵 A 和向量 b，然后用 scipy 中的共轭梯度法解方程得到融合的结果。

运行方法

```
python main.py --help
usage: main.py [-h] original_image_path mask_path patch_image_path output_directory

positional arguments:
  original_image_path  Path to the original image
  mask_path            Path to the mask image
  patch_image_path     Path to the patch image
  output_directory     Directory to save the result

optional arguments:
  -h, --help           show this help message and exit
```

例如：

```
python main.py ./data/completion/input4.jpg ./data/completion/input4_mask.jpg ./data/completion/input4_patch.jpg ./result
```

结果将保存在 result 文件夹中，命名方式为 input{i}_result.jpg。

耗时

生成一张图片耗时约为 1 分钟，大部分时间在求解方程上面。

结果

最左边是原图，旁边是未融合图像，再旁边是 80*80 的核，最右边是 1*1 的核。

参考的实现：

https://github.com/parosky/poissonblending/tree/master

https://github.com/Cydiater/scene-completion/tree/main