刘韬 2112404239 （homework4_lab03)

detect_keypoints():

1. 用 cv2.imread()读取图像
2. 使用 cv2.SIFT_create()检测关键点和描述符，后加 sift.detectAndCompute()

create_feature_matches()

1. 用 cv2.BFMatcher()和 matcher.knnMatch()来匹配特征
2. 迭代通过匹配对{m,n}，如果 m.distance<lowe_ratio*n.distance，则附加到要返回的 good_matches 中

create_ransac_match()

1. 调用 cv2.findEssentialMat()提取必需矩阵和 is_inlier 掩码

creve_scene_graph()

1. 循环通过每一组图像对{i,j}，并检查是否存在 numpy 文件为图像对使用 np.load()，如果有两个图像之间的对应，计算内部的数量和添加边节点 i 和 j，如果 np.load 抛出 OSError，连接下一个图像对。

get_init_image_ids()

1. 将变量 max_inliers 设置为 0，以跟踪当前最高数量的 inliers
2. 循环通过每个图像对{i,j}，使用 load_manths()获得匹配数
3. 如果数量匹配> max_inliers，请将其设置为新的 max_inliers 值，并覆盖以前的 max_pair，并将{i,j}设置为新的 max_pair

get_init_extrinsics()

1. 调用 cv2.recoverPose()在离化摄像机之间检索轮状矩阵 R 和横向向量 t
2. 调用 np.concatenate()，为一个照相机创建 3x4 投影矩阵[R|t]

get_next_pair()

1. 与 get_init_image_ids()类似，除了外环迭代注册的图像 idi，内环迭代相邻的节点 j，在 get_init_image_ids()的执行步骤 3 之前检查 j 没有注册

solve_pnp()

1. 调用 cv2.solvePnP()提取 R 和 tvec
2. 使用 cv2.Rodrigues()将 R 转换 rotation_mtx
3. 使用 get_reprojec)on_residuals()计算剩余残差

get_reprojec)on_residuals()

1. 将每个三维点转换为齐次坐标
2. 对于每个齐次三维点坐标，应用投影矩阵得到投影齐次二维坐标
3. 对于每个投影的均匀二维坐标，我们将 z 坐标归一化为 1
4. 对于每个归一化投影齐次二维坐标，我们取地面真值齐次二维坐标的差值，利用 np.linalg.nor()求出欧氏距离

add_points3d()

1. 用 triangulate()得到由三角坐标系计算出的三维坐标系

compute_ba_residuals()

1. 类似于 get_reprojec)的_残差（）

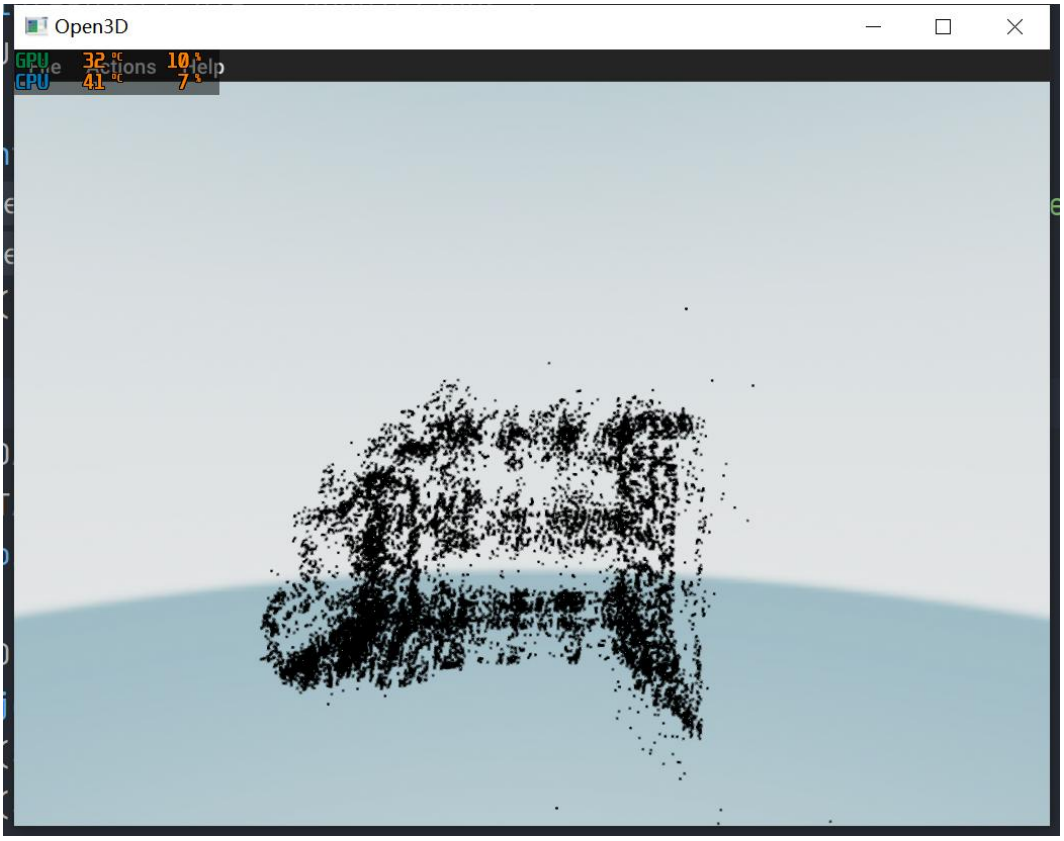

Fig1. Results of incremental SfM (no BA) for temple dataset

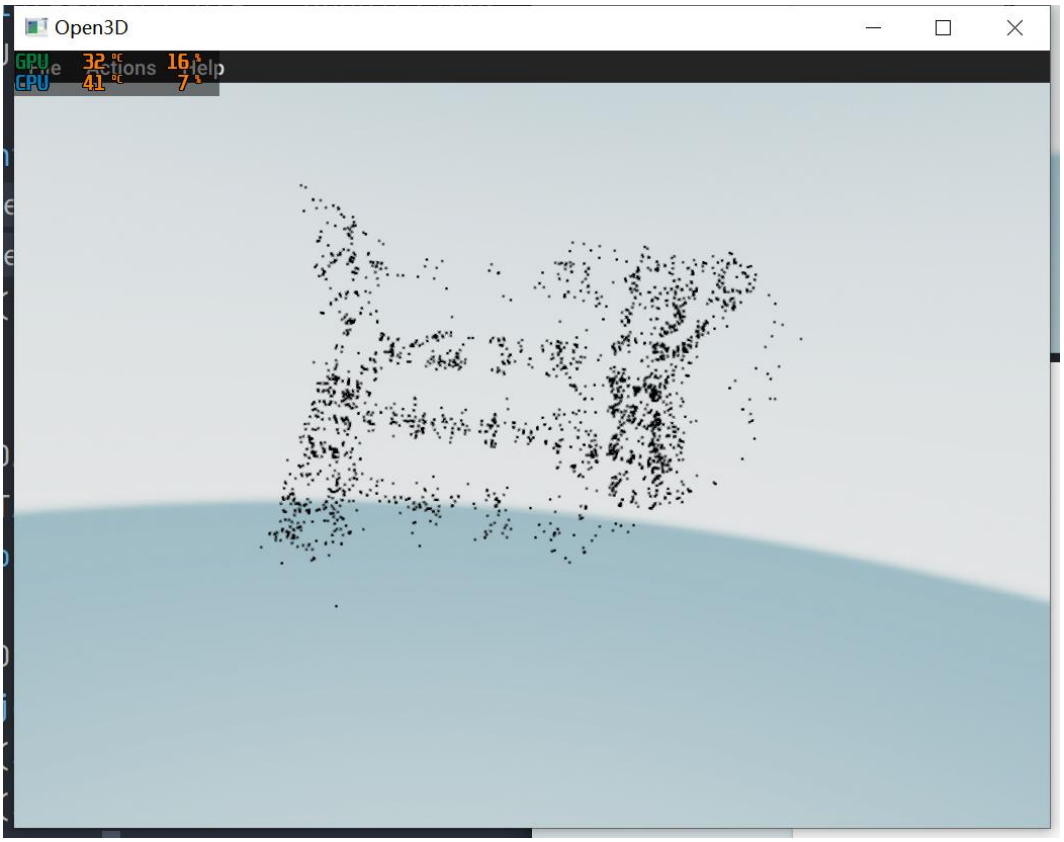

Fig2. Results of incremental SfM (no BA) for mini-temple dataset

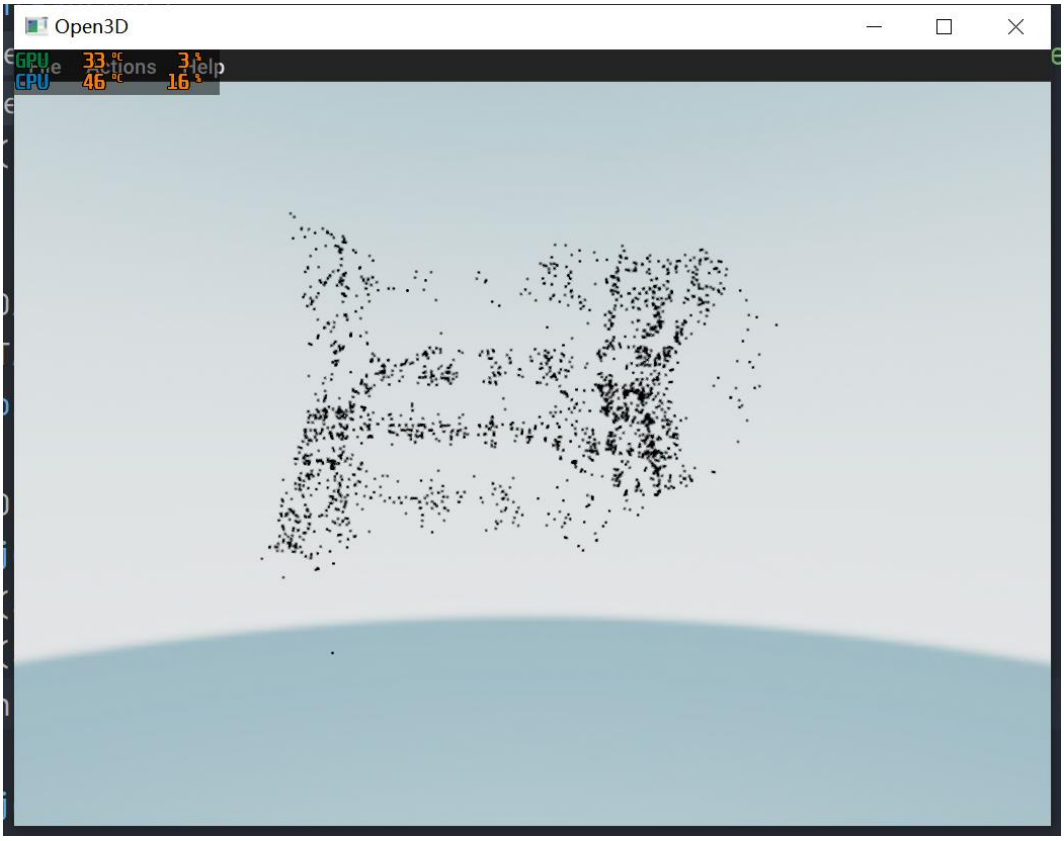

Fig3. Results of increamental SfM (with BA) for mini-temple dataset

参考：https://github.com/lihongguang0/CS4277_Labs