Report

1、预处理`preprocess.py`

主要用于执行运动结构（SfM）算法的预处理步骤。这个阶段的目标是从图像中检测关键点，匹配特征点，并为后续的三维重建和相机参数估计做好准备。

- **关键点检测**

脚本使用了 SIFT (Scale-Invariant Feature Transform) 算法来检测图像中的关键点。SIFT 特征具有尺度和旋转不变性，非常适合 SfM 任务中处理不同视角和光照变化的情况。具体实现中，首先通过 `cv2.SIFT_create()` 初始化 SIFT 检测器，然后使用 `detectAndCompute()` 来提取图像中的关键点和描述符。

- **特征匹配**

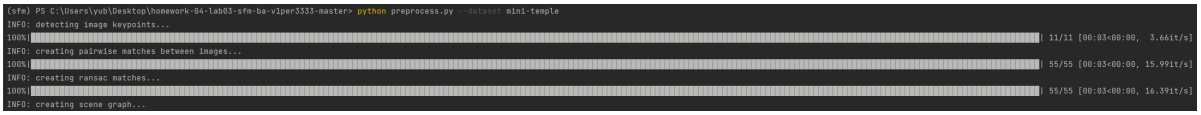
在特征匹配部分，使用了 OpenCV 提供的暴力匹配器 (BFMatcher) 进行特征匹配。为了提高匹配的准确性，采用了 Lowe 比例测试来过滤错误匹配，确保保留下来的特征点配对具有较高的可靠性。匹配后的结果保存在一个 `.npy` 文件中，用于后续的几何验证。

- **几何验证**

为了确保特征匹配的几何一致性，使用了 RANSAC 算法来计算两个图像之间的本质矩阵。通过 RANSAC 进行鲁棒估计，可以去除错误匹配的离群点，确保最终的特征匹配对能够用于精确的相机姿态估计。

- **输出**

在完成上述步骤后，`preprocess.py` 会将关键点及其描述符存储到 `.pkl` 文件中，同时保存特征匹配结果和经过 RANSAC 验证后的匹配对。这些输出将为后续的三维点重建和相机参数的优化提供基础。



2、`sfm.py` 代码

- **初始化**

初始化阶段通过选择图像对来计算相对姿态。选择的图像对是基于最大数量的RANSAC内点来确定的。选定的图像对用于计算其外参矩阵，包括旋转矩阵和位移向量。此外，通过三角测量 (Triangulation) 获取初始的三维点集。

- **三维点重建**

通过图像对中的匹配点，计算出三维点。这些匹配点通过图像的相机内参和外参矩阵进行投影，得到三维空间中的点云。

- **增量更新**

增量更新阶段通过不断注册新的图像，将新图像与已注册图像进行匹配，并通过PnP (Perspective-n-Point) 方法估计新图像的相对位姿。每当一个新图像被成功注册时，新的三维点会被重建，并与现有的三维点进行更新。此过程通过逐步连接图像对，确保场景重建的连贯性。

- **光束调整**

在整个增量式SFM过程中，为了优化相机的外参和三维点，采用了光束调整方法。光束调整通过最小化重投影误差，优化相机位姿和三维点的估计结果，进一步提高了重建的准确性。

get_init_image_ids

此函数通过遍历场景图，选择具有最多匹配内点的图像对作为初始化图像对。

get_init_extrinsics

假设初始化图像1的外参为[I | 0]，并通过计算图像对的本质矩阵，得到图像2的外参[R | t]。

- **triangulate****

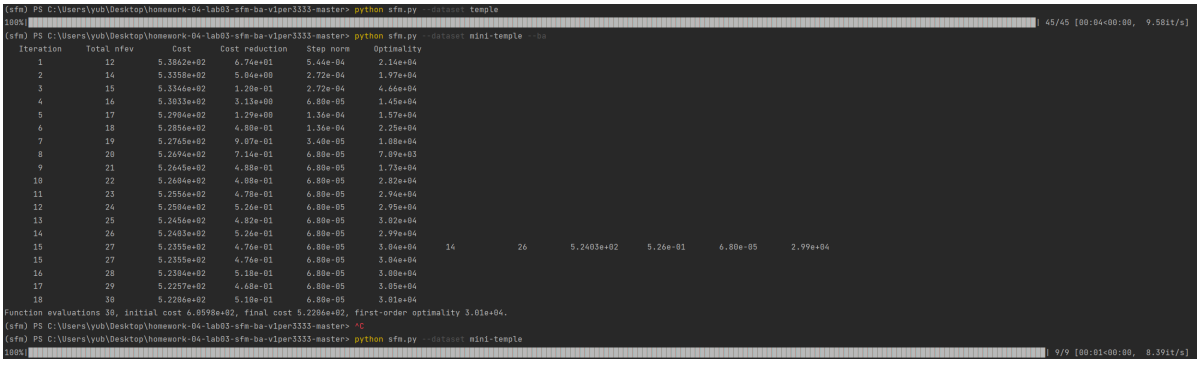
通过三角测量方法，计算出给定图像对和匹配点的三维坐标。

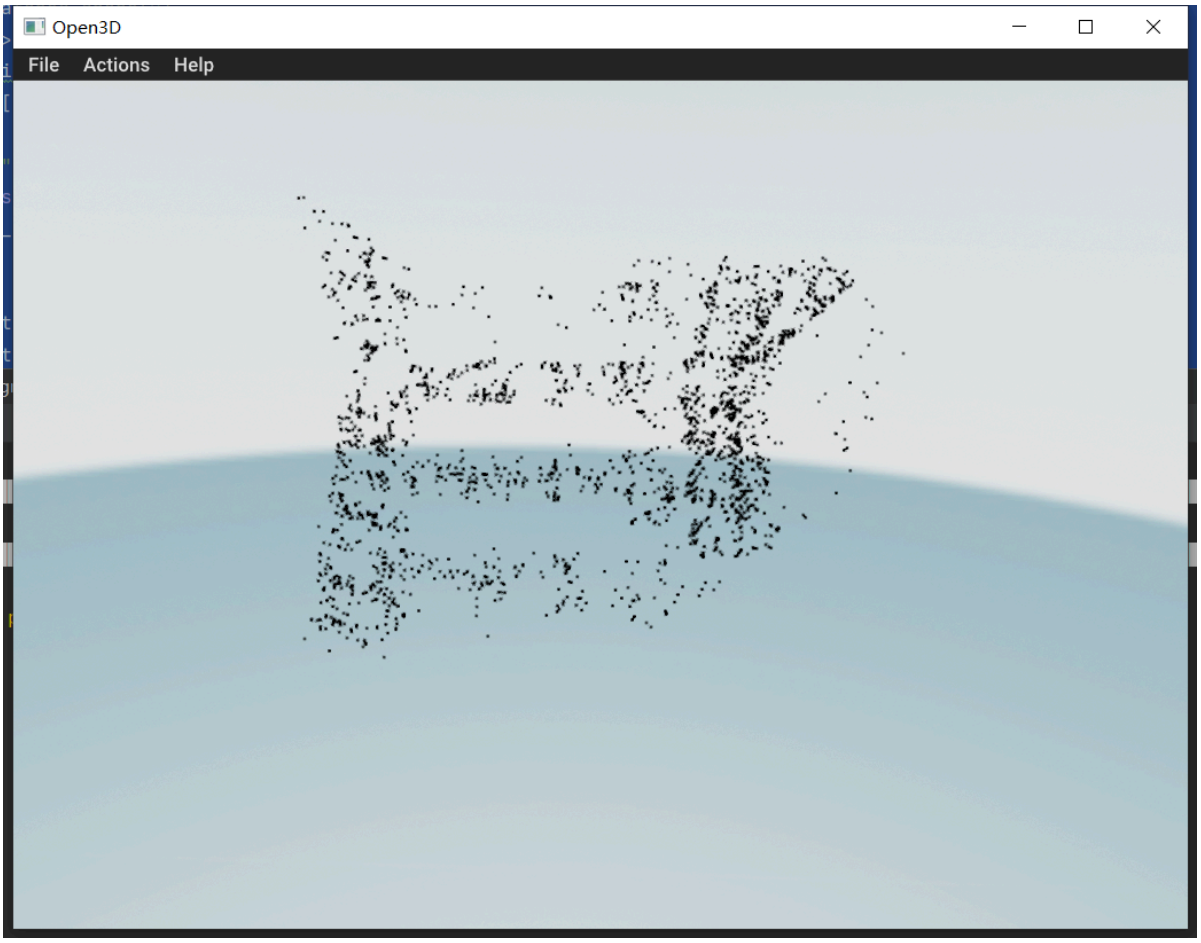
- **solve_pnp**

使用RANSAC算法解算PnP问题，估计出新的图像的旋转矩阵和位移向量。

- **bundle_adjustment**

该函数通过最小化光束调整残差，优化相机的内外参以及三维点的坐标，确保所有图像和三维点的相对位置更加精确。




Open3D
File
Actions
Help