姓名：曾铭生　学号：2112504379

1. 任务概述

本次作业实现了一个经典的增量式 SfM pipeline，主要包括：

1) 预处理：SIFT 关键点检测与描述子提取、特征匹配（Lowe ratio）、RANSAC 几何验证，并基于 inlier 数构建场景图；

2) SfM 初始化：从场景图中选取 inlier 数最多的初始图像对，通过本质矩阵恢复相对位姿并三角化得到初始 3D 点；

3) 增量式注册：从已注册集合出发迭代选择“新图像—已注册图像”对，利用 2D-3D 对应关系用带 RANSAC 的 PnP 求位姿，之后三角化新增 3D 点并更新对应关系；

4) Bundle Adjustment：以最小化所有观测的重投影误差为目标，用 `scipy.optimize.least_squares` 优化相机参数与 3D 点。

2. 预处理（preprocess.py）实现

2.1 SIFT 关键点检测：detect_keypoints

- 输入：单张图像路径
- 输出：保存该图像的 keypoints 与 descriptors 到 predictions/<dataset>/keypoints/<image_id>.pkl
- 方法：OpenCV cv2.SIFT_create() + detectAndCompute，对灰度图提取特征。

2.2 特征匹配 + Lowe Ratio：create_feature_matches

- 输入：图像对 (I1, I2)
- 方法：
  - cv2.BFMatcher().knnMatch(desc1, desc2, k=2) 得到每个特征的两近邻；
  - Lowe ratio test：若 d_1 < r \cdot d_2 则保留（本实现默认 r=0.6）；
- 输出：
  - 匹配索引 N×2（[i, j]）保存至 bf-match/；
  - 可视化图片保存至 bf-match-images/。

2.3 RANSAC 过滤：create_ransac_matches

- 输入：特征匹配结果
- 方法：
  - 优先调用 cv2.findEssentialMat(..., method=RANSAC) 得到本质矩阵 E 与 inlier mask；
  - 若环境缺少该 API，则 fallback：`findFundamentalMat` 得到 F，再用 $E = K^T F K$ 转换；
  - 仅保存 inlier 对应的匹配索引与 E。
- 输出：
  - ransac-match/：几何验证后的匹配索引
  - `ransac-fundamental/`：保存 E
  - ransac-match-images/：仅绘制 inlier 的连线可视化

2.4 场景图构建：create_scene_graph

- 原则：若某图像对的 RANSAC inlier 数 min_num_inliers，则在图中连边。
- 输出：scene-graph.json（邻接表）用于后续选择初始化对与增量注册。

3. SfM 初始化（sfm.py）实现
3.1 初始图像对选择：get_init_image_ids
- 在场景图的所有边中，读取对应 ransac-match 文件，选择 inlier 数最大的图像对作为初始化对。

3.2 初始外参估计：get_init_extrinsics
- 假设第一张相机位姿为 [I|0]；
- 从预处理阶段保存的本质矩阵 E 与匹配点对中恢复相对位姿：OpenCV recoverPose(E, pts1, pts2, K)；
- 得到第二张相机外参 [R|t]。

3.3 初始三角化
- 用 cv2.triangulatePoints 在两视图下对匹配点进行三角化，得到初始 3D 点云。

4. 增量式 SfM（sfm.py）实现
4.1 下一对选择：get_next_pair
- 从“已注册图像”出发，枚举其场景图邻居中“未注册图像”；
- 选择 RANSAC inlier 数最多的 (Inew, Ireg)。

4.2 带 RANSAC 的 PnP：solve_pnp
- 通过传递性建立 2D-3D 对应：
  - 已注册图像 Ireg 的 keypoint 与 3D 点存在对应关系；
  - Inew 与 Ireg 的匹配提供 Inew keypoint → Ireg keypoint；
  - 合并得到 Inew keypoint → 3D 点。
- RANSAC：每轮随机采样 6 个对应点，用 cv2.solvePnP(..., SOLVEPNP_ITERATIVE)求解 (R, t)，并以全体点的重投影误差判断 inliers。

4.3 重投影残差：get_reprojection_residuals
- 将 3D 点用 cv2.projectPoints 投影回像素坐标，残差为像素空间欧氏距离 \|u - \hat{u}\|_2。

4.4 新增 3D 点：add_points3d
- 对 (Inew, Ireg) 的匹配中，筛出 Ireg 上尚未绑定到任何 3D 点的 keypoints；
- 用两相机外参进行三角化生成新 3D 点并追加到点云；
- 更新两张图的 2D-3D 对应字典。

5. Bundle Adjustment（bundle_adjustment.py）实现
5.1 残差函数：compute_ba_residuals
- 输入参数向量包含：所有相机的 Rodrigues + t（每相机 6 维）以及所有 3D 点坐标；
- 目标：对每条观测 (camera_idx, point3d_idx, point2d) 计算重投影误差；
- 实现要求：无显式 for 循环，使用 NumPy 广播/矩阵乘。

6. 结果与可视化
- 重建运行：

  - python preprocess.py --dataset mini-temple
  - python sfm.py --dataset mini-temple
- 可视化：
  - python visualize.py --dataset mini-temple（若 BA 结果则加 --ba）