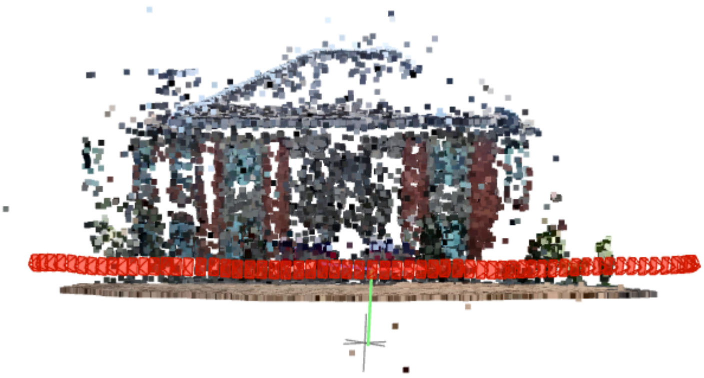
Structure from Motion and Relocalization

R14944054 李昊軒

Problem 1： COLMAP

COLMAP

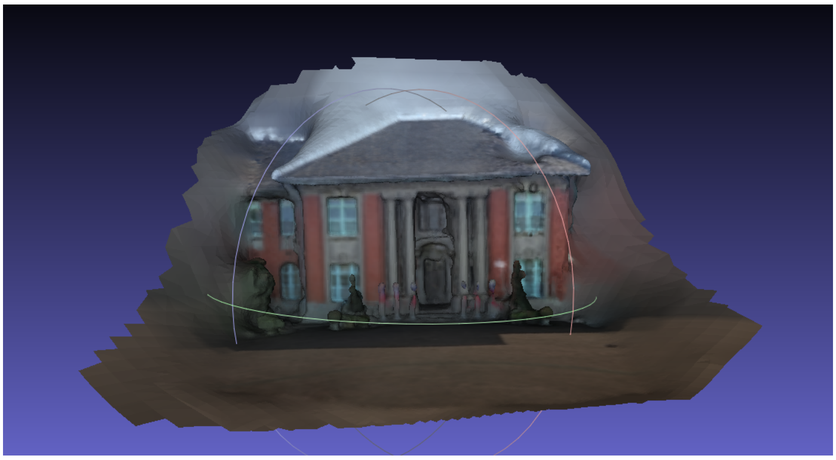
取出影片中的 Frames 並進行 Downsampling 以避免 COLMAP 運算過久，再將取出的 Downsampling 後的 Frames 丟入 COLMAP 進行 3D 重建建立稀疏點雲。



COLMAP Result Screenshot

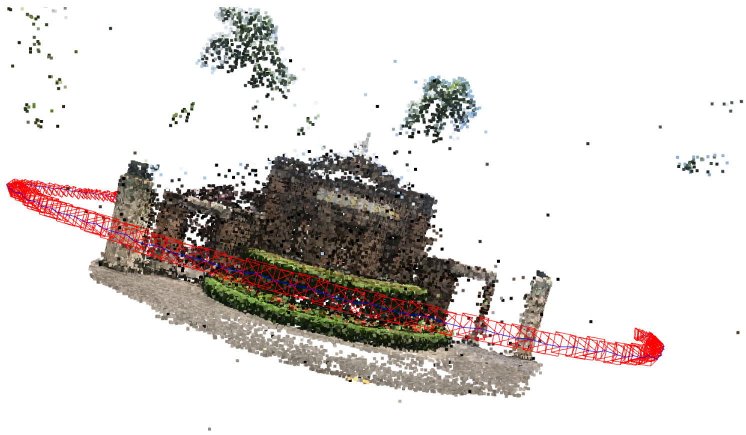
Meshlab

將點雲模型丟入 Meshlab 建立 Triangle Mesh Model，這裡使用 Screened Poisson Surface Reconstruction(Reconstruction Depth = 8, Minimum Number of Samples = 1.5, Interpolation Weights = 4)。



Meshlab Result Screenshot

Problem 2：Camera Relocalization



Relocation Screenshot

先將資料照影像名稱排好後，利用 3D 點雲與 Train Image 計算每個特徵點的平均 SIFT Descriptor 當作 Model，致使每個 3D 點雲的特徵點都有一個 Descriptor。接著將每張 Validation Image 的 Descriptor 與 Model Descriptor 進行配對 (在此嘗試了 Linear Search 和 K-D tree 兩種配對方式)，並以 Ratio = 0.75 進行 Ratio Test，保留好的配對。最後，利用 EPnP + RANSAC 求出旋轉與平移矩陣。

在這邊使用 EPnP + RANSAC 演算法 (max iter=2000, threshold=4.0, min sample=6, confidence=0.99)。首先，先將 2d 點利用 Distortion Coefficient 換算出未被扭曲的位置。接著執行 RANSAC 演算法，隨機取樣 min sample 個點用 EPnP 計算旋轉與平移矩陣並使用這兩個矩陣將 $\mathbf{P_{2d}}$ 轉換到世界座標並與 $\mathbf{P_{3d}}$ 計算距離當作誤差，最後挑選 Inliers 最多的結果。

Algorithm 1 PnP-RANSAC

Input: 3D points $\mathbf{P}_{3D}$, 2D points $\mathbf{P}_{2D}$, camera intrinsic matrix $\mathbf{K}$, distortion coefficients $\mathbf{d}$ (optional), reprojection threshold τ, maximum iterations $N_{\max}$, sample size m, confidence η
Output: Estimated rotation $\mathbf{R}^*$, translation $\mathbf{t}^*$, and inlier set $\mathcal{I}^*$
if $\mathbf{d}$ is not None **then**
    $\mathbf{P}_{2D} \leftarrow \text{UndistortPoints}(\mathbf{P}_{2D}, \mathbf{K}, \mathbf{d})$
end if
Initialize $\mathcal{I}^* \leftarrow \emptyset$, $\mathbf{R}^*, \mathbf{t}^* \leftarrow \text{None}$
for $i = 1$ to $N_{\max}$ **do**
    S = Randomly sample m indices
    $(\mathbf{R}, \mathbf{t}) \leftarrow \text{EPnP}(\mathbf{P}_{3D}[\mathcal{S}], \mathbf{P}_{2D}[\mathcal{S}], \mathbf{K})$
    Compute reprojection errors $e_j = \text{ReprojError}(\mathbf{P}_{3D}[j], \mathbf{P}_{2D}[j], \mathbf{R}, \mathbf{t}, \mathbf{K})$
    $\mathcal{I} = \{j \mid e_j < \tau\}$
    if $|\mathcal{I}| > |\mathcal{I}^*|$ **then**
        $\mathcal{I}^* \leftarrow \mathcal{I}$, $\mathbf{R}^* \leftarrow \mathbf{R}$, $\mathbf{t}^* \leftarrow \mathbf{t}$
        Compute inlier ratio $w = |\mathcal{I}|/N$
        Clamp w to $[10^{-6}, 1 - 10^{-6}]$
        $k_{\text{needed}} = \left\lceil \frac{\log(1-\eta)}{\log(1-w^m)} \right\rceil$
        $N_{\max} \leftarrow \min(N_{\max}, k_{\text{needed}})$
    end if
    if $i \geq N_{\max}$ **then**
        break
    end if
end for
$\mathbf{R}^* \leftarrow \text{Rodrigues}(\mathbf{R}^*)$
return $(\text{True}, \mathbf{R}^*, \mathbf{t}^*, \mathcal{I}^*)$

在 EPnP 中，首先會先計算出所有 3D 點的 Centroid 以其當作第一個 Control Point，接著用 SVD 計算出三個主要維度並加上 Centroid 形成其他三個 Control Point。以這些 Control Point 計算出他們的 Barycentric Coordinates $\boldsymbol{\alpha}$，並利用 $\boldsymbol{\alpha}$、相機內參與 2D 點座標形成矩陣 M 解 M 之 Null Space。

根據 Null Space 維度以不同方式求解 $\boldsymbol{\beta}$ 並對 $\boldsymbol{\beta}$ 做 Refinement，再利用 $\boldsymbol{\beta}$ 計算出相機座標之 Control Point，當 Null Space 維度大於 4 時則直接取最後一個維度當作 Control Point。

最後，依據得到的相機座標 Control Point 與是世界座標之 Control Point，使用 Arun's Method 得出旋轉與平移矩陣。

Algorithm 2 Efficient Perspective-n-Point (EPnP) Algorithm

Input: 3D points $\mathbf{P}_{3D} = \{\mathbf{X}_i\}_{i=1}^{N}$, 2D image points $\mathbf{P}_{2D} = \{\mathbf{x}_i\}_{i=1}^{N}$, camera intrinsic matrix $\mathbf{K}$
Output: Estimated rotation $\mathbf{R}$ and translation $\mathbf{t}$
Extract (f_x, f_y, c_x, c_y) from $\mathbf{K}$
Compute centroid $\boldsymbol{\mu}_w = \frac{1}{N}\sum_i \mathbf{X}_i$
Define world control points: $\mathbf{C}_w^{(0)} = \boldsymbol{\mu}_w$
Perform SVD on centered 3D points: $[\mathbf{U}, \mathbf{S}, \mathbf{V}^T] = \mathrm{SVD}(\mathbf{P}_{3D} - \boldsymbol{\mu}_w)$
Set:

$$\mathbf{C}_w^{(1)} = \boldsymbol{\mu}_w + \mathbf{V}_1\sigma_x, \quad \mathbf{C}_w^{(2)} = \boldsymbol{\mu}_w + \mathbf{V}_2\sigma_y, \quad \mathbf{C}_w^{(3)} = \boldsymbol{\mu}_w + \mathbf{V}_3\sigma_z$$

Compute barycentric coordinates $\boldsymbol{\alpha}_i = [\alpha_{i1}, \alpha_{i2}, \alpha_{i3}, \alpha_{i4}]$ s.t.

$$\mathbf{X}_i = \sum_{j=1}^{4} \alpha_{ij}\mathbf{C}_w^{(j)}$$

Construct projection matrix $\mathbf{M} \in \mathbb{R}^{2N\times 12}$
for $i = 1$ to N **do**
    $(u_i, v_i) \leftarrow \mathbf{x}_i$
    Fill rows:

$$\mathbf{M}_{2i} = [\alpha_{i1}f_x, 0, \alpha_{i1}(c_x - u_i), \ldots]$$
$$\mathbf{M}_{2i+1} = [0, \alpha_{i1}f_y, \alpha_{i1}(c_y - v_i), \ldots]$$

end for
Compute SVD of $\mathbf{M}$: $[\mathbf{U}, \mathbf{S}, \mathbf{V}^T] = \mathrm{SVD}(\mathbf{M})$
Determine null space $\mathbf{V}_{\text{null}}$ where singular values $< 10^{-8}$
Let $N_{\text{null}} = |\mathbf{V}_{\text{null}}|$, and reshape each vector into 4×3 control points V_{ctrl}
Compute world distances:

$$\rho_k = \|\mathbf{C}_w^{(a)} - \mathbf{C}_w^{(b)}\|^2, \quad \forall (a, b) \in \text{All Control Point Pairs}$$

if $N_{\text{null}} = 1$ **then**
    Estimate $\beta = \frac{\sum \|v_a - v_b\|\|w_a - w_b\|}{\sum \|v_a - v_b\|^2}$
else if $N_{\text{null}} = 2$ **then**
    Build linear system $\mathbf{L}\boldsymbol{\beta} = \rho$, where $\mathbf{L}_k = [v_1^2, 2v_1v_2, v_2^2]$
    Solve by pseudo-inverse: $\boldsymbol{\beta} = \mathbf{L}^+\rho$
else if $N_{\text{null}} = 3$ **then**
    Construct $\mathbf{L} \in \mathbb{R}^{6\times 6}$ using all pair combinations of $\mathbf{V}_{ctrl}^{(1:3)}$
    Solve $\boldsymbol{\beta} = \mathbf{L}^+\rho$
else
    $\boldsymbol{\beta} = [1]$
end if

Algorithm 3 Efficient Perspective-n-Point (EPnP) Algorithm Cont.

Refine $\boldsymbol{\beta}$ using nonlinear least-squares:

$$\boldsymbol{\beta} \leftarrow \arg\min_{\beta} \|\text{epnp\_error}(\beta, \mathbf{V}_{ctrl}, \mathbf{C}_w)\|^2$$

Reconstruct camera control points:

$$\mathbf{C}_c = \sum_i \beta_i \mathbf{V}_{ctrl}^{(i)}$$

Compute 3D points in camera frame:

$$\mathbf{P}_c = \boldsymbol{\alpha}\mathbf{C}_c$$

Compute centroids:

$$\bar{\mathbf{P}}_c = \text{mean}(\mathbf{P}_c), \quad \bar{\mathbf{P}}_w = \text{mean}(\mathbf{P}_w)$$

Center data and compute correlation matrix:

$$\mathbf{H} = (\mathbf{P}_w - \bar{\mathbf{P}}_w)^T(\mathbf{P}_c - \bar{\mathbf{P}}_c)$$

Apply SVD: $[\mathbf{U}, \mathbf{S}, \mathbf{V}^T] = \text{SVD}(\mathbf{H})$
$\mathbf{R} = \mathbf{V}\mathbf{U}^T$
if $\det(\mathbf{R}) < 0$ **then**
    Flip sign of last column of $\mathbf{V}$, recompute $\mathbf{R} = \mathbf{V}\mathbf{U}^T$
end if
$\mathbf{t} = \bar{\mathbf{P}}_c - \mathbf{R}\bar{\mathbf{P}}_w$
return $\mathbf{R}, \mathbf{t}$

將旋轉矩陣以 quaternion 方式表示並以以下方式計算旋轉誤差：

$$R_{\text{rel}} = \begin{bmatrix} w_1w_2 + x_1x_2 + y_1y_2 + z_1z_2 \\ w_1x_2 - x_1w_2 + y_1z_2 - z_1y_2 \\ w_1y_2 - x_1z_2 - y_1w_2 + z_1x_2 \\ w_1z_2 + x_1y_2 - y_1x_2 - z_1w_2 \end{bmatrix}$$

$$\theta = 2 \arctan\left(\frac{\sqrt{x^2 + y^2 + z^2}}{|w|}\right)$$

平移矩陣則以以下方式計算：

$$t_{\text{error}} = \|\mathbf{t}_1 - \mathbf{t}_2\|$$

最後，將每張 Validation Image 的旋轉與平移矩陣轉換為 4 * 4 矩陣 c2w，計算其反矩陣並將相機投影回真實世界座標。

Comparison

Matching Method Comparison

Matching Method	Linear Search	K-D tree
Rotation Error	0.1847	0.1847
Translation Error	0.0091	0.0091
Time	298.7534	280.2897

上表為使用 Linear Search 與 K-D Tree 進行特徵點配對的結果。由上表可知，在不損失 Error 的情況下，利用 K-D Tree 能夠減少運算時間。

Sample Count Comparison

Sample Number	4	6
Rotation Error	164.0545	0.1847
Translation Error	6.3036	0.0091

上表為 P4P 與 P6P 的結果，由上表可知，P6P 比 P4P 大幅增加準確率，代表 4 個點難以精準預測 Camera Pose。

Beta Comparison

β	Using different dimensions of the null space	$\beta = 1$
Rotation Error	0.1847	3.2697
Translation Error	0.0091	3.4038

上表為 EPnP 中，最後 beta 計算根據 Null Space 維度進行不同運算與直接取 $\beta = 1$ 的結果比較。可以看出 β 經過 Null Space 不同維度運算後能夠減少誤差。

Dependencies

- Python 3.11.10
- Numpy 1.26.4
- OpenCV 4.10.0
- Open3d 0.19.0
- Scipy 1.13.1
- Pandas 2.3.3

Usage

```
$ python3 2d3dmathcing.py
```

Language Model

- ChatGPT

Youtube Link

https://youtu.be/NYkpTAWyixA