Lab 2-1

Young Min Kim

2022.07.14.

3D LAB VISION

Table of Contents

1. ICP (Iterative Closest Point) Algorithm - Points to Points

1. Implement ICP with Python Code

- In this example, we will implement ICP algorithm with Python Code.
- Given the two scans of the Stanford bunny (`lab2-1/data/bun000.ply`, `bun045.ply`), align two scans with ICP algorithm.
- You need to write a function `find_correspondence` and `find_registration` in `lab2-1/utils.py`

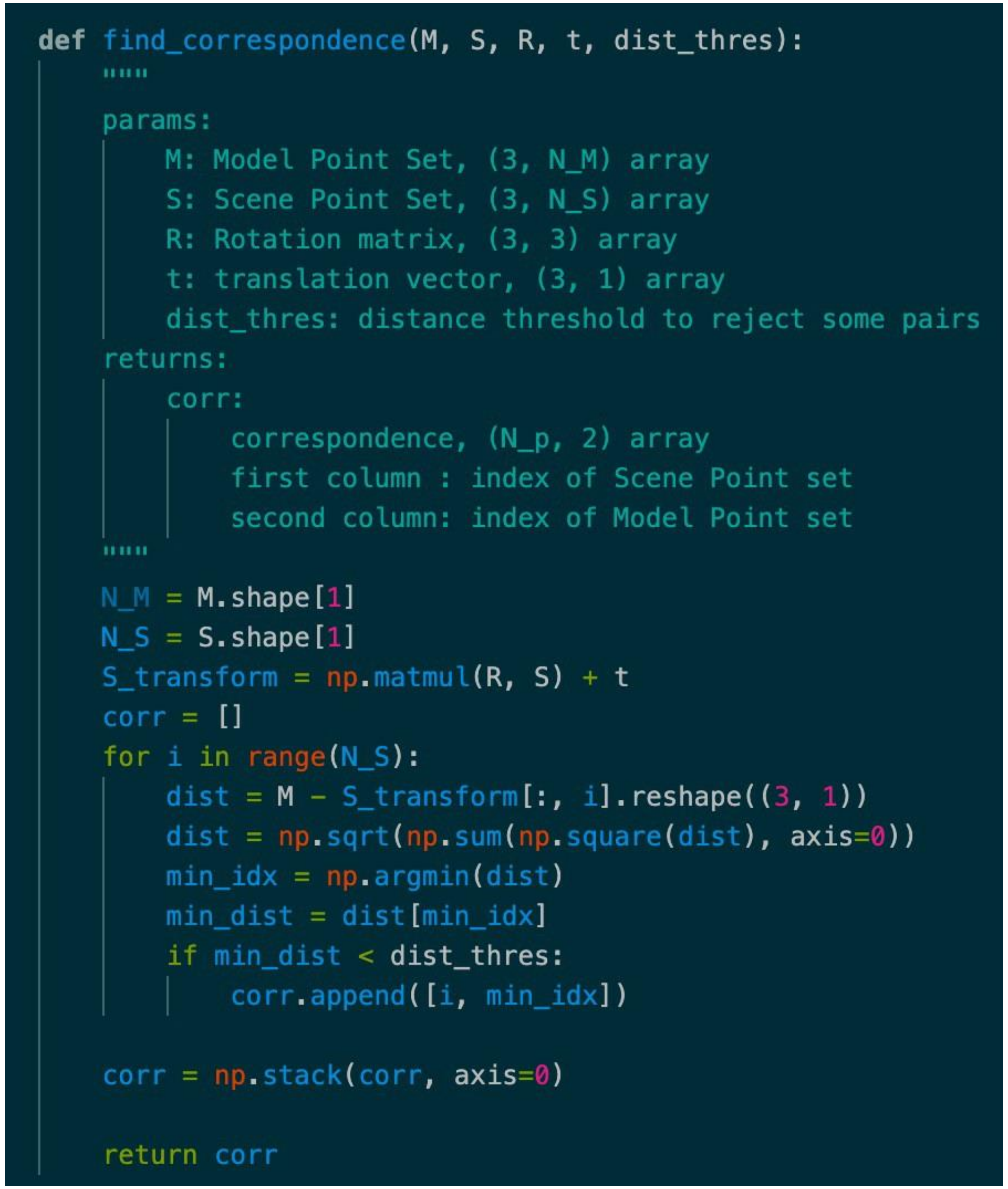
- Try rejecting some of the correspondences with different distance threshold.
- Plot MSE-iteration graph.

2. Implement ICP with Python Code

- Finding Correspondence (closest point)
  - $d(s, M) = \min_{m \in M} d(m, s)$

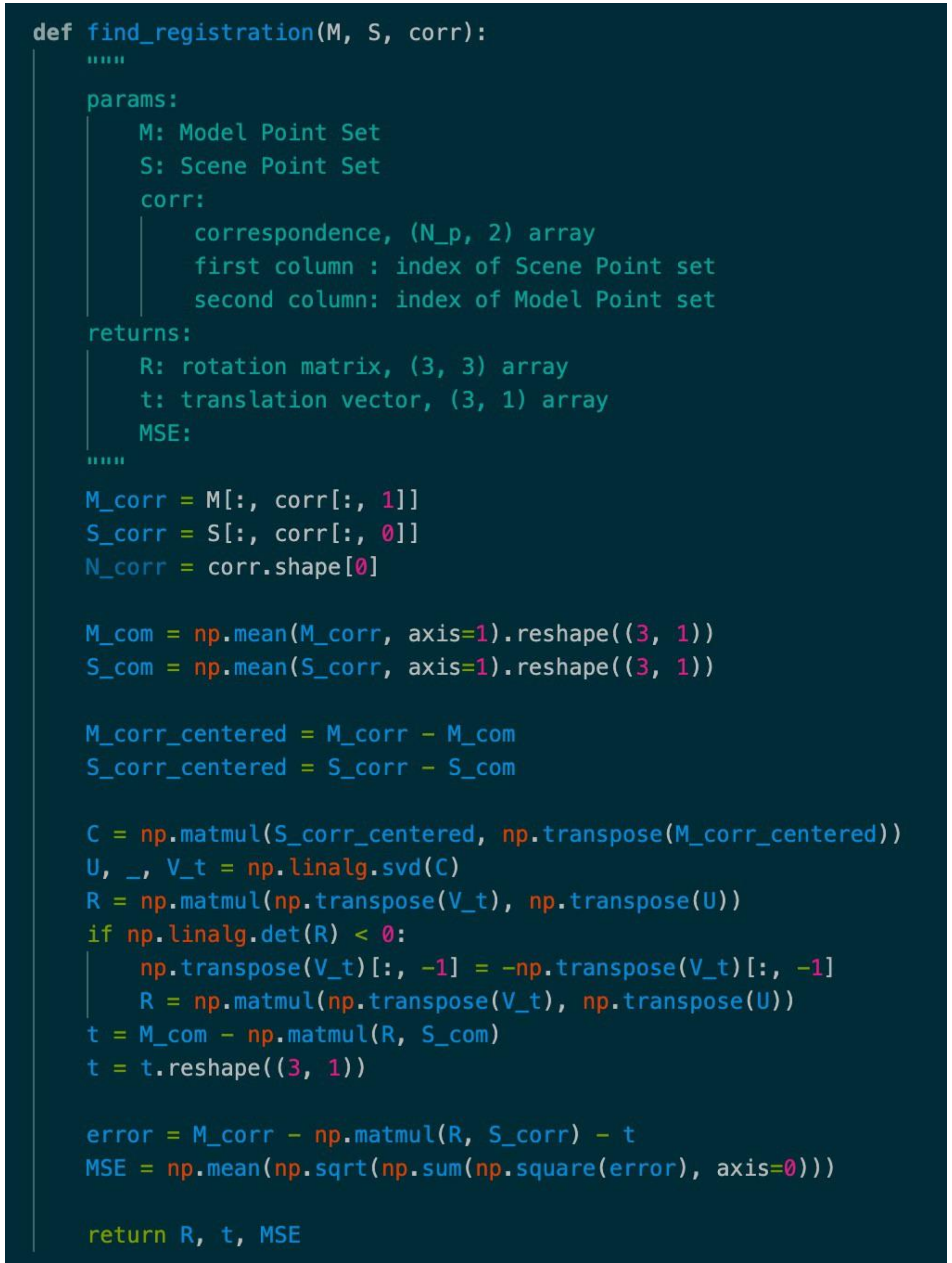
- Corresponding set alignment with MSE objective
  - $M' = M - \mu_M$
  - $S' = S - \mu_S$
  - $C = S'M = U\Sigma V^T$
  - $\rightarrow R = VU^T, t = \mu_M - R\mu_S$

2. Implement ICP with Python Code



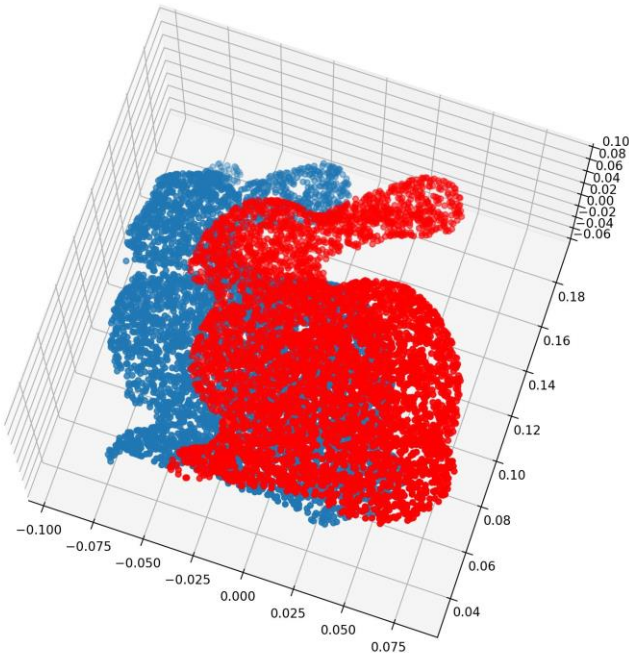
```python
def find_correspondence(M, S, R, t, dist_thres):
    """
    params:
        M: Model Point Set, (3, N_M) array
        S: Scene Point Set, (3, N_S) array
        R: Rotation matrix, (3, 3) array
        t: translation vector, (3, 1) array
        dist_thres: distance threshold to reject some pairs
    returns:
        corr:
            correspondence, (N_p, 2) array
            first column : index of Scene Point set
            second column: index of Model Point set
    """
    N_M = M.shape[1]
    N_S = S.shape[1]
    S_transform = np.matmul(R, S) + t
    corr = []
    for i in range(N_S):
        dist = M - S_transform[:, i].reshape((3, 1))
        dist = np.sqrt(np.sum(np.square(dist), axis=0))
        min_idx = np.argmin(dist)
        min_dist = dist[min_idx]
        if min_dist < dist_thres:
            corr.append([i, min_idx])

    corr = np.stack(corr, axis=0)

    return corr
```

2. Implement ICP with Python Code

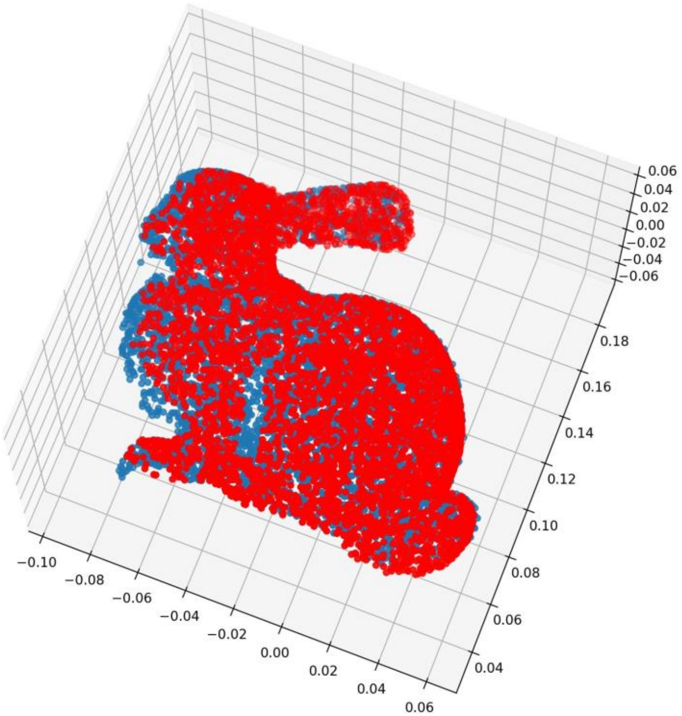


```python
def find_registration(M, S, corr):
    """
    params:
        M: Model Point Set
        S: Scene Point Set
        corr:
            correspondence, (N_p, 2) array
            first column : index of Scene Point set
            second column: index of Model Point set
    returns:
        R: rotation matrix, (3, 3) array
        t: translation vector, (3, 1) array
        MSE:
    """
    M_corr = M[:, corr[:, 1]]
    S_corr = S[:, corr[:, 0]]
    N_corr = corr.shape[0]

    M_com = np.mean(M_corr, axis=1).reshape((3, 1))
    S_com = np.mean(S_corr, axis=1).reshape((3, 1))

    M_corr_centered = M_corr - M_com
    S_corr_centered = S_corr - S_com

    C = np.matmul(S_corr_centered, np.transpose(M_corr_centered))
    U, _, V_t = np.linalg.svd(C)
    R = np.matmul(np.transpose(V_t), np.transpose(U))
    if np.linalg.det(R) < 0:
        np.transpose(V_t)[:, -1] = -np.transpose(V_t)[:, -1]
        R = np.matmul(np.transpose(V_t), np.transpose(U))
    t = M_com - np.matmul(R, S_com)
    t = t.reshape((3, 1))

    error = M_corr - np.matmul(R, S_corr) - t
    MSE = np.mean(np.sqrt(np.sum(np.square(error), axis=0)))

    return R, t, MSE
```

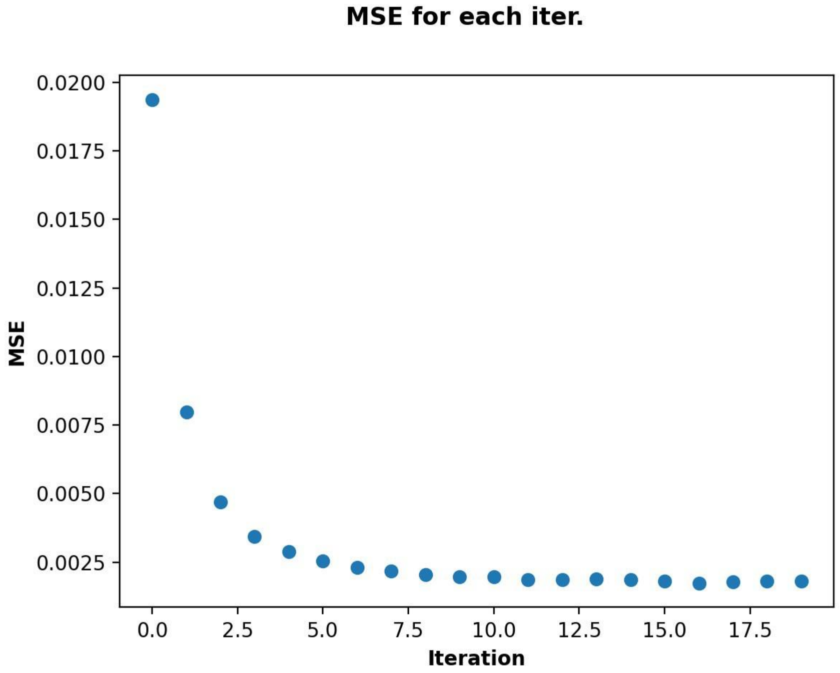
2. Implement ICP with Python Code



Before Allignment



After Allignment with ICP



2. Implement ICP with Python Code

- Experiment with different `dist_thres` and different iteration number.