16-833 SLAM HW4: Dense SLAM with Point-based Fusion

Ryan Wu (Andrew ID: weihuanw)

April 17, 2024

1 Overview

Dataset downloaded and processed.

2 Iterative Closest Point (ICP)

2.1 Projective data association

The conditions u, v, d must satisfy to set a valid correspondence is given by the following:

$$\begin{bmatrix} 0 \leq u < W \\ 0 \leq v < H \\ 0 \leq d \end{bmatrix}$$

The additional distance and angle threshold filter is necessary for better and more accurate association between the point cloud and the vertex map. The distance filter threshold makes sure the most likely points are matched and the outliers are properly filtered out. The angle thresholds can help associate the points that are similar in orientation. Overall, this threshold will improve the overall match of the point clouds.

2.2 Linearization

The rewrite of r_i is given by the following:

$$r_i(\alpha, \beta, \gamma, tx, ty, tz) = A_i \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ tx \\ ty \\ tz \end{bmatrix} + b_i, \text{ where } (i = 1, 2, 3)$$

where A_i is:

$$\begin{bmatrix} -n_{q2}P'_3 + n_{q3}P'_2 \\ n_{q1}P'_3 - n_{q3}P'_1 \\ -n_{q1}P'_2 + n_{q2}P'_1 \\ n_{q1} \\ n_{q2} \\ n_{q3} \end{bmatrix}$$

and b_i is:

$$\begin{bmatrix} n_{q1}(P'_1 - q_1) + n_{q2}(P'_2 - q_2) + n_{q3}(P'_3 - q_3) \end{bmatrix}$$

2.3 Optimization

The linear system that provides a closed-form solution of $(\alpha, \beta, \gamma, t_x, t_y, t_z)$ in terms of A_i and b_i using LU formulation is given by the following:

$$\delta = (A^T A)^{-1} A^T b$$

where A is:

$$A = \sum_{i=1}^{n} A_i^T A_i$$

and b is:

$$b = \sum_{i=1}^{n} A_i^T b_i$$

The registration output for the default source and target (frame 10 and 50) case was successful and both the average loss and the inlier counts converged. This is because these frames may have more overlaps and proper correspondences. The complexity and variation in each frame are less dramatic, which results in proper point cloud alignments and convergence.

In the case of a more challenging source and target (frame 10 to 100), the registration failed because the scenes' complexity hindered the ICP registration performance. The more point clouds' variance in distance and angle makes the matching process more computationally heavy. With only 10 iterations, the system was unable to converge properly. Furthermore, there might be more noise in each frame which can also cause convergent failure.

Figure 1: Point clouds before and after registration (frame 10 and 50).

Figure 2: Point clouds before and after registration (frame 10 and 100).

3 Point-based Fusion

3.1 Filter

The mask array implementation was successful and the results from the following sections further confirm.

3.2 Merge

The weighted average of the position is given by the following:

$$W_{\text{position}} = \frac{w(p) + R_c^w(q) + t_c^w}{w+1}$$

The weighted average of the normals is given by the following:

$$W_{\text{normals}} = \frac{w(n_p) + R_c^w(n_q)}{w+1}$$

3.3 Addition

The add function implementation was successful and the results from the following sections further confirm.

3.4 Results

Figure 3: Fusion with ground truth poses with a normal map.

The final number of points on the map is 1,363,469. Furthermore, the compression ratio is around 8.355%.

4 The Dense SLAM System

For our dense SLAM system, the source is the input RGBD-frame and the target is the map being built and updated. The roles can be swapped depending on the use case and desired outcomes. However, one must consider the sensor properties and the unique data transformations in the framework.

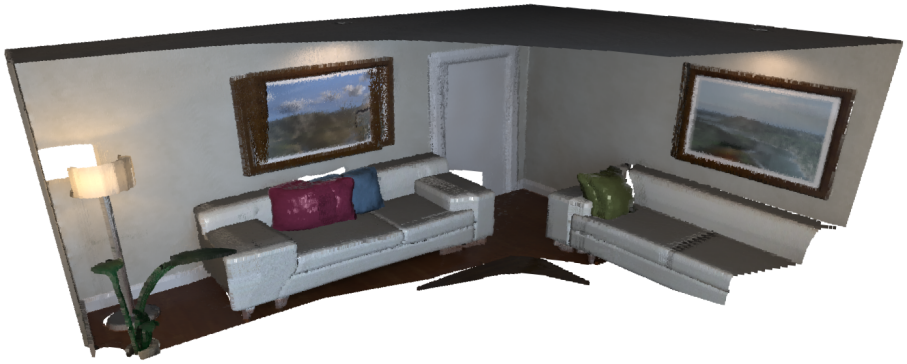

Figure 4: The output visualization of my dense SLAM system implementation.

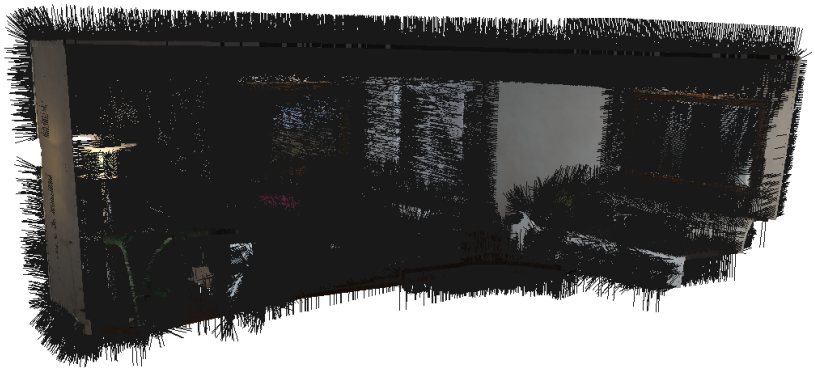

Figure 5: The output visualization of my dense SLAM system implementation (normal).

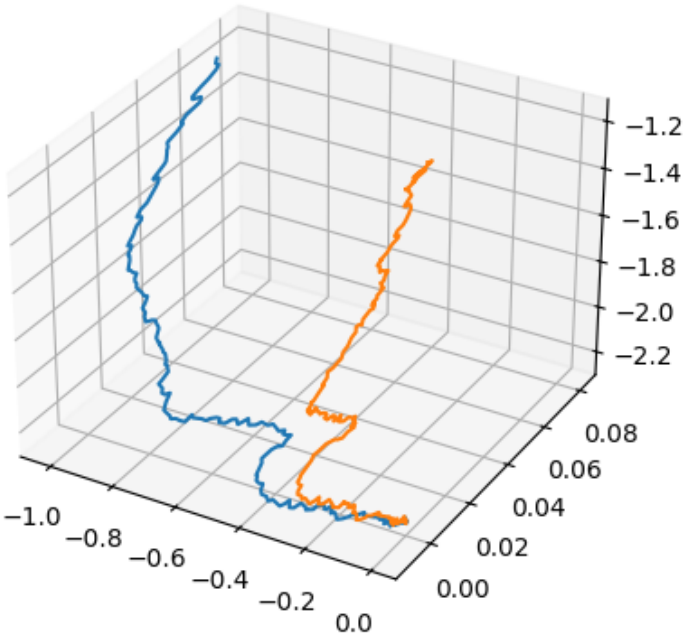

Figure 6: The system's recorded estimated trajectory and the ground truth trajectory.

5 Code Submission Rules

Code submitted on Gradescope.

References

[1] Keller, Maik, et al. "Real-time 3D reconstruction in dynamic scenes using point-based fusion." International Conference on 3D Vision (3DV), 2013. `http://ieeexplore.ieee.org/document/6599048/`.

[2] Newcombe, Richard A., et al. "KinectFusion: Real-time dense surface mapping and tracking." 10th IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 2011. `http://ieeexplore.ieee.org/document/6162880/`.

[3] Handa, Ankur, et al. "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM." IEEE International Conference on Robotics and Automation (ICRA), 2014. `https://www.doc.ic.ac.uk/~ahanda/VaFRIC/iclnuim.html`.

[4] Sola, Joan and Deray, Jeremie and Atchuthan, Dinesh. "A micro Lie theory for state estimation in robotics," arXiv 2018.

16-833 SLAM HW4 Dense SLAM with Point-based Fusion

2 Iterative Closest Point (ICP)

2.2 Linearization

Given:

$$\delta R = \begin{bmatrix} 1 & -\gamma & \beta \\ \gamma & 1 & -\alpha \\ -\beta & \alpha & 1 \end{bmatrix}, \quad \delta t = [t_x, t_y, t_z]$$

$$\sum_{i \in \Omega} r_i^2(\delta R, \delta t) = \left\| n_{q_i}^T \left((\delta R) p_i' + \delta t - q_i \right) \right\|^2, \quad p_i' = R^0 p_i + t^0$$

Find:

$$r_i(\alpha, \beta, \gamma, t_x, t_y, t_z) = A_i \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ t_x \\ t_y \\ t_z \end{bmatrix} + b_i$$

A_i is 6×1

b_i is scalar

Solution:

$$r_i(\delta R, \delta t) = n_{q_i}^T \left((\delta R) p_i' + \delta t - q_i \right) \rightarrow r_i(\delta R, \delta t) = n_{q_i}^T \left((\delta R)(R^0 p_i + t^0) + \delta t - q_i \right)$$

$$r_i(\delta R, \delta t) = n_{q_i}^T \left(\begin{bmatrix} 1 & -\gamma & \beta \\ \gamma & 1 & -\alpha \\ -\beta & \alpha & 1 \end{bmatrix} p_i' + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} - q_i \right) \quad \text{where } i = (1, 2, 3)$$

$$r_i(\delta R, \delta t) = [n_{q_1} \; n_{q_2} \; n_{q_3}] \left(\begin{bmatrix} 1 & -\gamma & \beta \\ \gamma & 1 & -\alpha \\ -\beta & \alpha & 1 \end{bmatrix} \begin{bmatrix} p_1' \\ p_2' \\ p_3' \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} - \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} \right)$$

$$r_i(\delta R, \delta t) = [n_{q_1} \; n_{q_2} \; n_{q_3}] \begin{bmatrix} p_1' - \gamma p_2' + \beta p_3' + t_x - q_1 \\ \gamma p_1' + p_2' - \alpha p_3' + t_y - q_2 \\ -\beta p_1' + \alpha p_2' + p_3' + t_z - q_3 \end{bmatrix} = \begin{bmatrix} n_{q_1}(p_1' - \gamma p_2' + \beta p_3' + t_x - q_1) \\ n_{q_2}(\gamma p_1' + p_2' - \alpha p_3' + t_y - q_2) \\ n_{q_3}(-\beta p_1' + \alpha p_2' + p_3' + t_z - q_3) \end{bmatrix}$$

$$r_i(\alpha, \beta, \gamma, t_x, t_y, t_z) = \begin{bmatrix} -n_{q_2} p_3' + n_{q_3} p_2' \\ n_{q_1} p_3' - n_{q_3} p_1' \\ -n_{q_1} p_2' + n_{q_2} p_1' \\ n_{q_1} \\ n_{q_2} \\ n_{q_3} \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ t_x \\ t_y \\ t_z \end{bmatrix} + \left[n_{q_1}(p_1' - q_1) + n_{q_2}(p_2' - q_2) + n_{q_3}(p_3' - q_3) \right]$$

A_i (6×1)

b_i (scalar)

2.3 Optimization

Given:

$$\sum_{i=1}^{n} r_i^2(\alpha, \beta, \gamma, t_x, t_y, t_z) = \sum_{i=1}^{n} \left\| A_i \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ t_x \\ t_y \\ t_z \end{bmatrix} + b_i \right\|^2$$

Find: write down the linear system that provides a closed form solution

Solution:

$$\text{delta} = (A^T A)^{-1} A^T b \quad , \quad A = \sum_{i=1}^{N} A_i^T A_i \quad , \quad b = \sum_{i=1}^{N} A_i^T b_i \ \#$$

Note: use -b in Python solvers.

3 Point-based Fusion

3.2 Merge

Given: $p \in \mathbb{R}^3$, w, $q \in \mathbb{R}^3$, $w=1$

Find: the weight average of the positions in terms of p, q, R_c^w, t_c^w, w

the weight average of normals in terms of n_p, n_q, R_c^w, w

Equation: $$w = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i}$$

Solutions: $w_i = (w+1)$

$$W_{positions} = \frac{w(p) + 1[R_c^w(q) + t_c^w]}{w+1} \quad \#$$

$$W_{normals} = \frac{w(n_p) + 1[R_c^w(n_q)]}{w+1} \quad \#$$

3.4 Results

Given: width = 680, height = 480, # of total points = 1,363,469

Find: the compression ratio

Equation: $$CR = \frac{\#\text{ of total points}}{(H/2 \cdot W/2 \cdot 200)}$$

Solution:

$$CR = \frac{1,363,469}{\left(\frac{480}{2} \cdot \frac{680}{2} \cdot 200\right)} \approx 0.08355 \text{ or } 8.355\% \quad \#$$