Homework 4: ICP Fusion

Christopher Klammer
cklammer@andrew.cmu.edu

April 15, 2023

1 Iterative Closest Point

1.1 Projective Data Association

1.1.1 Conditions for First Filter (5 Points)

When projecting the source points onto the target points:

- The projected source points must lie on the target image plane:
  - $u \geq 0$
  - $v \geq 0$
  - $u \leq H$
  - $v \leq W$
- The projected source points must be in front of the camera:
  - $d(u, v) > 0$

1.1.2 Second Filter Discussion (5 Points)

The second filter will see if the corresponding 3D points are close enough. Essentially, this is to see if the correspondence is valid. This is necessary because when there are large angles between frames and potential parallax, this check essentially makes sure that the small angle assumption is witheld. Otherwise, We could get two points that have very different depth values and would likely not yield a good solution because it is outside the assumption.

1.2 Linearization (15 Points)

1.2.1 Expansion

$$p_i' = R^0 p_i + t^0$$

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

$$r_i(\delta R, \delta t) = ||n_{q_i}^T(\begin{bmatrix} 1 & -\gamma & \beta \\ \gamma & 1 & -\alpha \\ -\beta & \alpha & 1 \end{bmatrix} \begin{bmatrix} (p')_i^x \\ (p')_i^y \\ (p')_i^z \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} - \begin{bmatrix} q_i^x \\ q_i^y \\ q_i^z \end{bmatrix})$$

$$r_i(\delta R, \delta t) = n_{q_i}^T(\begin{bmatrix} (p')_i^x - \gamma(p')_i^y + \beta(p')_i^z + t_x - q_i^x \\ (p')_i^x\gamma + (p')_i^y - \alpha(p')_i^z + t_y - q_i^y \\ -\beta(p')_i^x + \alpha(p')_i^y + (p')_i^z + t_z - q_i^z \end{bmatrix})$$

$$r_i(\delta R, \delta t) = n_{q_i}^T(\begin{bmatrix} (p')_i^x - \gamma(p')_i^y + \beta(p')_i^z + t_x - q_i^x \\ (p')_i^x\gamma + (p')_i^y - \alpha(p')_i^z + t_y - q_i^y \\ -\beta(p')_i^x + \alpha(p')_i^y + (p')_i^z + t_z - q_i^z \end{bmatrix})$$

$$r_i(\delta R, \delta t) = n_{q_i}^x * ((p')_i^x - \gamma(p')_i^y + \beta(p')_i^z + t_x - q_i^x) + n_{q_i}^y * ((p')_i^x\gamma + (p')_i^y - \alpha(p')_i^z + t_y - q_i^y) + n_{q_i}^z * (-\beta(p')_i^x + \alpha(p')_i^y$$

1.2.2 Formulate into $r_i = Ax + b$

Variables in A:

$$n_{q_i}^x(-\gamma(p')_i^y + \beta(p')_i^z + t_x)$$

$$n_{q_i}^y((p')_i^x\gamma - \alpha(p')_i^z + t_y)$$

$$n_{q_i}^z(-\beta(p')_i^x + \alpha(p')_i^y + t_z)$$

$$A = \begin{bmatrix} -n_{q_i}^y(p')_i^z + n_{q_i}^z(p')_i^y \\ n_{q_i}^x(p')_i^z - n_{q_i}^z(p')_i^x \\ -n_{q_i}^x(p')_i^y + n_{q_i}^y(p')_i^x \\ n_{q_i}^x \\ n_{q_i}^y \\ n_{q_i}^z \end{bmatrix}^T$$

Variables in b:

$$b_1 = n_{q_i}^x((p')_i^x - q_i^x)$$
$$b_2 = n_{q_i}^y((p')_i^y - q_i^y)$$
$$b_3 = n_{q_i}^z((p')_i^z - q_i^z)$$

Final Form:

$$r_i = \begin{bmatrix} -n_{q_i}^y(p')_i^z + n_{q_i}^z(p')_i^y & n_{q_i}^x(p')_i^z - n_{q_i}^z(p')_i^x & -n_{q_i}^x(p')_i^y + n_{q_i}^y(p')_i^x & n_{q_i}^x & n_{q_i}^y & n_{q_i}^z \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ t_x \\ t_y \\ t_z \end{bmatrix} + (b_1 + b_2 + b_3)$$

As the pdf alludes to, we can use the cross-product matrix to simplify the derivation

$$[p']_\times = \begin{bmatrix} 0 & -(p')_i^z & (p')_i^y \\ (p')_i^z & 0 & -(p')_i^x \\ -(p')_i^y & (p')_i^x & 0 \end{bmatrix}$$

$$C = [p']_\times \begin{bmatrix} n_{q_i}^x \\ n_{q_i}^y \\ n_{q_i}^z \end{bmatrix}$$

The resulting matrix is a $3x1$ matrix, so, we can extract each row to be an entry in our final matrix. Let's call those entries C_1, C_2, C_3 for clarity.

$$r_i = \begin{bmatrix} C_1 & C_2 & C_3 & n_{q_i}^x & n_{q_i}^y & n_{q_i}^z \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ t_x \\ t_y \\ t_z \end{bmatrix} + (b_1 + b_2 + b_3)$$

1.3 Optimization

1.3.1 Linear System: QR Factorization (15 Points)

I chose to use QR factorization where each row of A is a projected point correspondence. From the previous part, we can expand our formulation to be a matrix instead of a vector by stacking the vectors:

Notation:

$\vec{n^d} = \{n_{q_i}^d; \forall i \in q\}; \forall d \in \{x, y, z\}$

$\vec{n^d} \in \mathbb{R}^M$

$\vec{C_{1:3}} \in \mathbb{R}^M$

Additional Formulation:

$$\begin{bmatrix} \vec{C_1} & \vec{C_2} & \vec{C_3} & n^x & n^y & n^z \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ t_x \\ t_y \\ t_z \end{bmatrix} + (\vec{b_1} + \vec{b_2} + \vec{b_3}) = \vec{0}$$

$$\begin{bmatrix} \vec{C_1} & \vec{C_2} & \vec{C_3} & \vec{n^x} & \vec{n^y} & \vec{n^z} \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ t_x \\ t_y \\ t_z \end{bmatrix} = -(\vec{b_1} + \vec{b_2} + \vec{b_3})$$

$$QR = A = \begin{bmatrix} \vec{C_1} & \vec{C_2} & \vec{C_3} & \vec{n^x} & \vec{n^y} & \vec{n^z} \end{bmatrix}$$

$$x = R^{-1} \cdot (Q^T \cdot b)$$

1.3.2 Visualization (10 Points)

Figure 1: Frames 10 and 50 Before Optimization

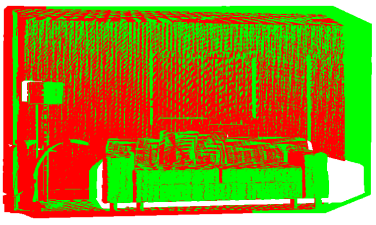

Figure 2: Frames 10 and 50 After Optimization

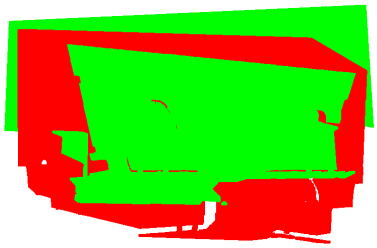

Figure 3: Frames 150 and 300 Before Optimization

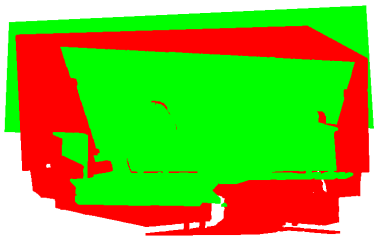

Figure 4: Frames 150 and 300 After Optimization

Analysis The frames 10 and 50 are able to be aligned very well because of the small angle assumption. That is, there is not much rotation between the two frames. However, in frames 150 and 300, we can see that there is a rather large rotation between the two frames and some parallax effects. For example, see the bottom left and top left corner of the wall, there is a large rotation between those two points. Whereas, for frames 10 and 50, they are extremely close and the rotation is much smaller so the assumption is more valid in this case.

2 Point Based Fusion

2.1 Filter (5 Points)

2.1.1 Filter 1

- $u \geq 0$
- $v \geq 0$
- $u \leq H$
- $v \leq W$

2.1.2 Filter 2

- $arccos(\frac{n_q \cdot n_p}{||n_q|| ||n_p||}) \leq T_{angle}$
- $||p_p - p_q|| \leq T_{dist}$

See code writeup for more details.

2.2 Merge (15 Points)

2.2.1 Weighted Average of Positions

$$q_{world} = R_c^w q + t_c^w$$
$$p = \frac{q_{world} + w * p}{w + 1}$$

2.2.2 Weighted Average of Colors

$$p_c = \frac{q_c + w * p_c}{w + 1}$$

2.2.3 Weight Average of Normals

$$n_{q_{world}} = R_c^w n_q$$
$$n_p = norm(\frac{w * n_p + n_{q_{world}}}{w + 1})$$

2.2.4 Updating Weights

$$w[idx] = w + 1$$

2.2.5 Compression Ratio

$$Ratio = \frac{N}{(frame_T - frame_0) * H * W}$$
$$N = len(p_{map})$$

Number Points: 1362157
Compression Rate: 0.08912773502931323

2.2.6 Addition (10 Points)

Those entries that are new are concatenated to the numpy arrays with $w = 1$. See the code for more additional details.

2.3 Results (10 Points)

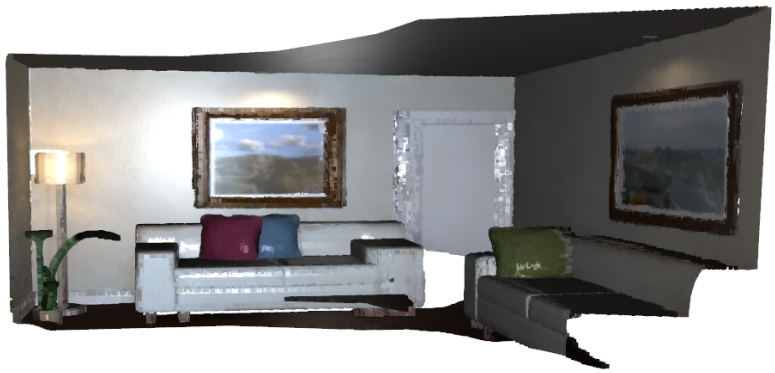

Figure 5: Visualization of Point-Based Map with Colors

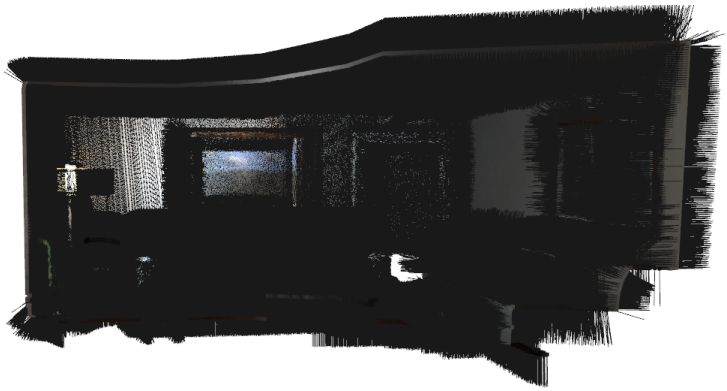

Figure 6: Visualization of Point-Based Normal Map

3 Dense SLAM

3.1 Source and Target (5 Points)

The source here is the map and the target is the RGB-D frame. I don't really think a swap is possible because when looking for valid correspondence, we need to pass the filter for correspondence which assumes we are projecting the 3D points to a 2D plane. If the points are not in front of the image plane or on the image plane, they would be filtered out. If you swapped the role, when performing ICP, you would need to take the points on the image and compare them to all of the points in 3D, for one this would be extremely expensive, two, nothing would be theoretically filtered out in filter one and, in general, would require a completely different strategy if this behavior was desired. If there was something more logical like a binning spatial representation of the 3D points, this could possibly be more plausible but would not work for our current setup.

3.2 Visualization (10 Points)

3.2.1 Results

Figure 7: Visualization of Dense Map with Colors

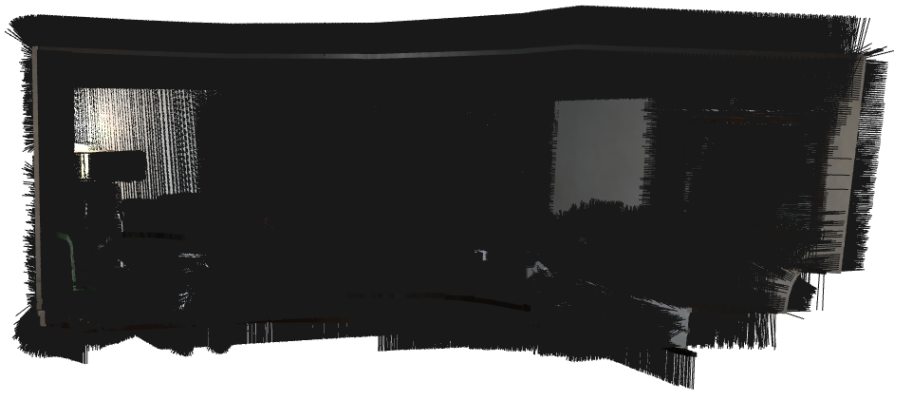

Figure 8: Visualization of Dense Normal Map

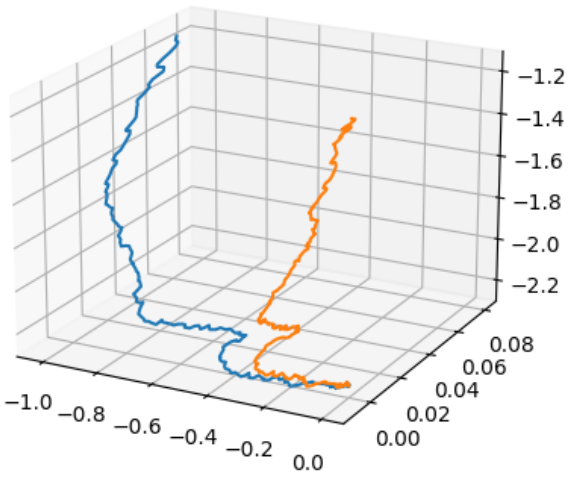

Figure 9: Trajectory of the Camera in Dense SLAM

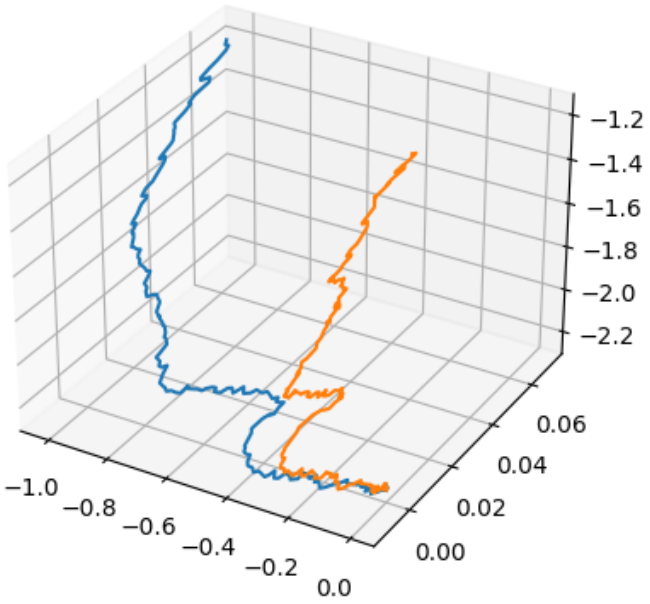

Figure 10: Trajectory after odometry and pruning

3.2.2 Reducing Drift (Extra Credit: 10 Points)

I chose to do the following:

1. If points were not updated in the past 35 timesteps, remove them to mitigate drift and spurious associations

2. Use the built in Open3D RGB-D odometry to receive a more accurate initial pose estimation before performing ICP

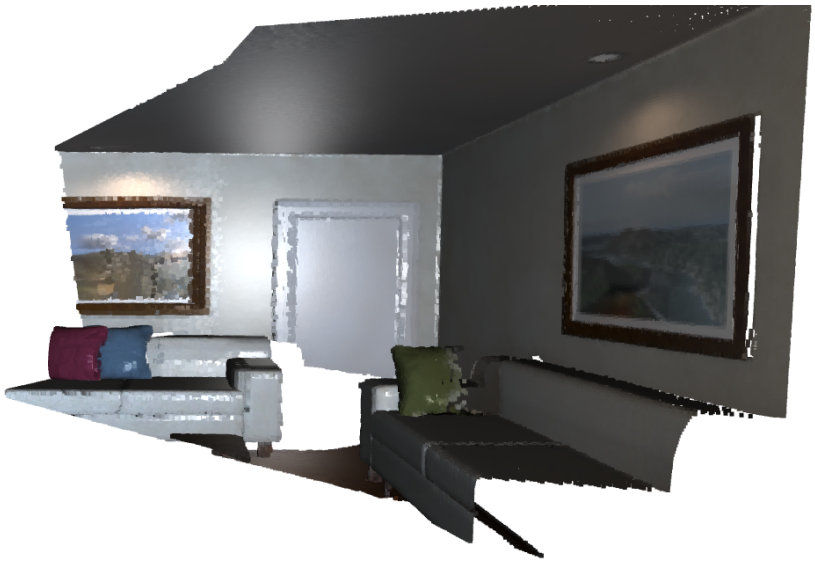

Figure 11: Color after odometry and pruning

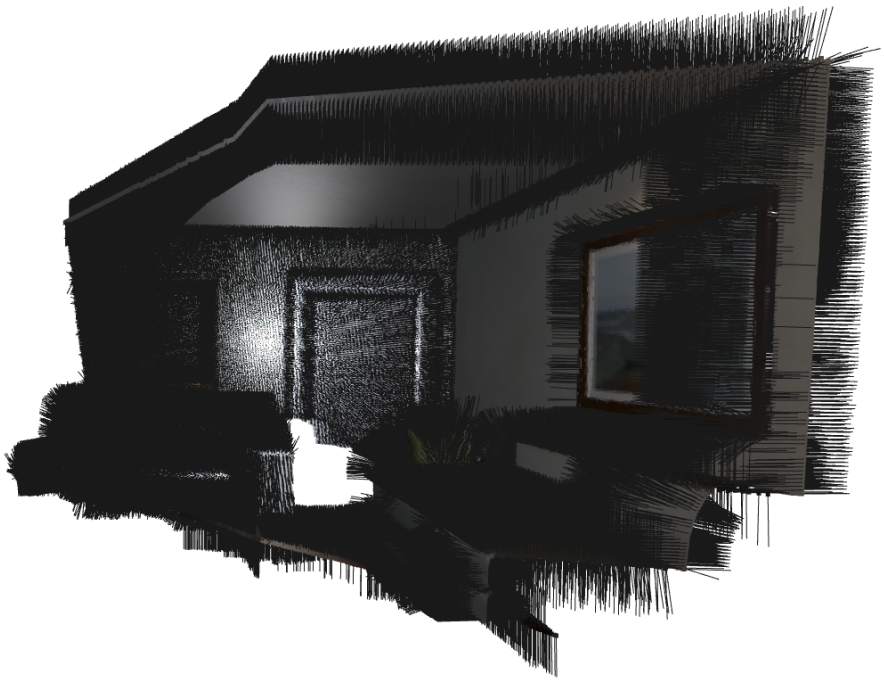

Figure 12: Normal after odometry and pruning