Abhay Gupta (abhayg) **Homework 4** April 18, 2019

Iterative Closest Point (ICP)

(a) Point-to-point ICP defines an energy function to reduce the Euclidean distance between the current point and the point in a subsequent timestep that has undergone some transformation; yielding an estimate of a transform. **Point-to-plane ICP** defines an energy function to minimize the dot product of a normal vector estimate and the difference between a vertex and a transformed vertex at a subsequent time step. This also yields an estimate of a transform, with an emphasis on maintaining surfaces.

(b) We want to estimate $\tilde{T}_{g,k}$ which transforms points in the camera frame to the world frame and consequently provides a camera pose. To estimate this new transform, we compute multiple iterations of small changes to the transform and solve for a gradient at each iteration.

At each iteration z, define $\tilde{T}^z_{inc}$ such that,

$$\tilde{T}^z_{g,k} = \tilde{T}^z_{inc} * \tilde{T}^{z-1}_{g,k}$$

Since we are making the small angle assumption, we get the following incremental transformation:

$$\tilde{T}^z_{inc} = \left[\tilde{R}^z \mid \tilde{t}^z\right] = \begin{bmatrix} 1 & \alpha & -\gamma & t_x \\ -\alpha & 1 & \beta & t_y \\ \gamma & -\beta & 1 & t_z \end{bmatrix}$$

We can represent the incremental transformation as a parameter vector

$$\tilde{T}^z_{inc} = \begin{bmatrix} \beta \\ \gamma \\ \alpha \\ t_x \\ t_y \\ t_z \end{bmatrix}$$

To derive this representation, let us set up the minimization by equating the current frame transform estimate applied to a camera frame vertex to the incremental transformation applied to the world frame vertex.

$$\tilde{T}^z_{g,k}\dot{V}_k = \tilde{R}^z\tilde{V}^g_k(u) + \tilde{t}^z$$

Let us represent $\tilde{V}^g_k(u) = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$. Then,

$$
\begin{aligned}
\tilde{T}^z_{g,k}\dot{V}_k &= \begin{bmatrix} v_1 + \alpha v_2 + (-\gamma v_3) \\ -\alpha v_1 + v_2 + \beta v_3 \\ \gamma v_1 + (-\beta v_2) + v_3 \end{bmatrix} + \tilde{t}^z \\
&= \begin{bmatrix} 0 + \alpha v_2 + (-\gamma v_3) \\ -\alpha v_1 + 0 + \beta v_3 \\ \gamma v_1 + (-\beta v_2) + 0 \end{bmatrix} + \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} + \tilde{t}^z \\
&= \begin{bmatrix} 0 - \gamma v_3 + \alpha v_2 \\ \beta v_3 + 0 - \alpha v_1 \\ -\beta v_2 + \gamma v_1 + 0 \end{bmatrix} + \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} + \tilde{t}^z \\
&= [\tilde{V}^g_k(u)]_\times \begin{bmatrix} \beta \\ \gamma \\ \alpha \end{bmatrix} + \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} + \tilde{t}^z \\
&= [\tilde{V}^g_k(u)]_\times \mid I_{3\times 3}] \begin{bmatrix} \beta \\ \gamma \\ \alpha \\ t_x \\ t_y \\ t_z \end{bmatrix} + \tilde{V}^g_k(u) \\
&= G(u)x + \tilde{V}^g_k(u)
\end{aligned}
$$

This now allows solving incremental transform parameters using linear solvers.

(c) Code is implemented in the corresponding files and is available in the `ICP_fusion/` folder.

(d) Three views from the result of `is_debug_icp = 1` are given below in Figure 1. After viewing at different angles, we see that point-to-plane ICP performs well. It is able to register the points coherently and form a properly structured output with good connectivity between different surfaces of the 3D figure. We, however, notice that the trajectory is not well-formed in this case and breaks off quite far from the end position.

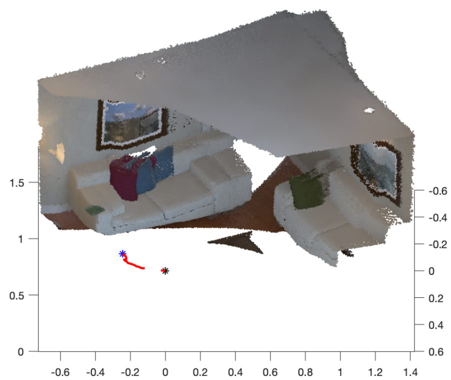
Point-based Fusion

(a) Volumetric based fusion methods have a large computational overhead involved due to continuous changes in data representations over different modules. Also, a regular voxel grid imposes a large memory overhead. The memory represents both empty spaces and surfaces densely and thus limits the size of the reconstruction volume.

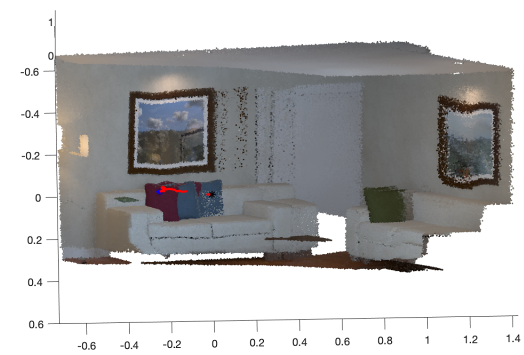
(b) Consider a point $\mathbf{p}$ mapped into a new frame to point $\mathbf{p}$' by rotation R and translation $\mathbf{t}$, then, we have $\mathbf{p}' = R\mathbf{p} + \mathbf{t}$. Given the rotation matrix R and the normal vecor $\mathbf{n}$, the new transformed normal **n'** is given by

$$\mathbf{n}' = R\mathbf{n}$$

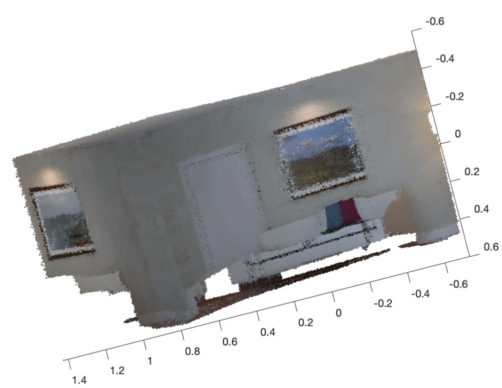
(c) Code is implemented in the corresponding files and is available in the `ICP_fusion/` folder.



(a) View 1



(b) View 2



(c) View 3

Figure 1: Results of ICP Fusion

(d) Three views from the result of `is_debug_icp = 0` are given below in Figure 2-4. After viewing at different angles, we see that point-based fusion does not perform as well as ICP in terms of structured outputs. It is not only able to register the points coherently but also extract the exact shape of the object to a good extent. We, also, notice that the trajectory is well formed in this case and is connected to the end.

Results and Discussions

(a) Using the fusion model as the next reference frame results in a better ICP registration than using the current frame as the next reference frame. This is because of the way the fusion model is computed. It accounts not only for direct transforms between the different iterations, but it also accounts for the fact that between different transforms, we have the closeness of the projected points and its corresponding input points and also that they are in a similar normal direction.

(b) The Root Mean Squared Error (RMSE) measures how far the computed transform estimate is from the actual transform estimate. The lower the value of the RMSE, the more accurate the computed transform and

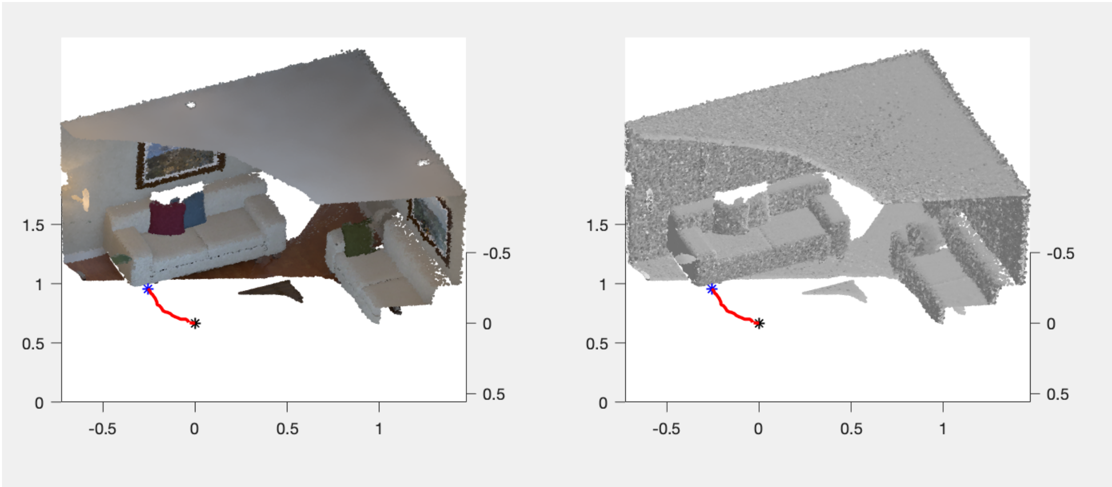


Figure 2: Point-Based Fusion - View 1

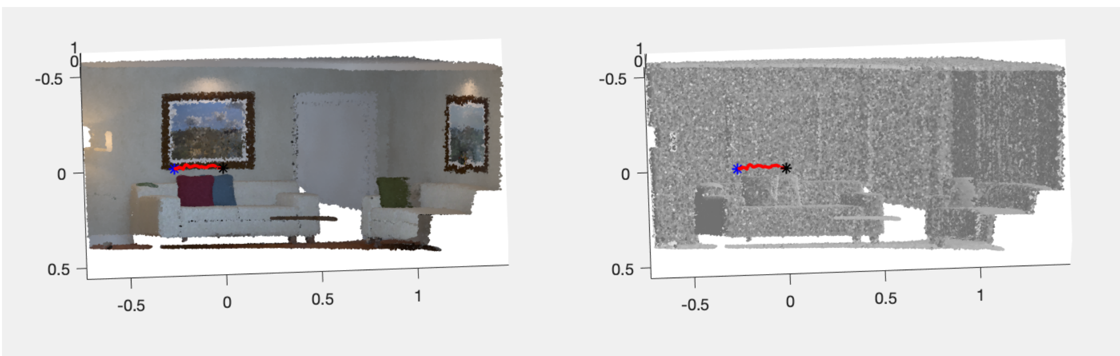


Figure 3: Point-Based Fusion - View 2

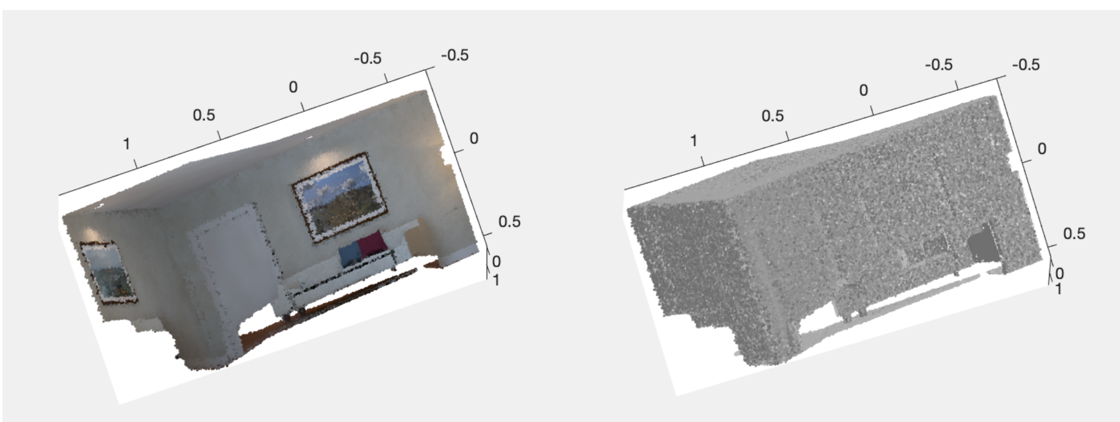
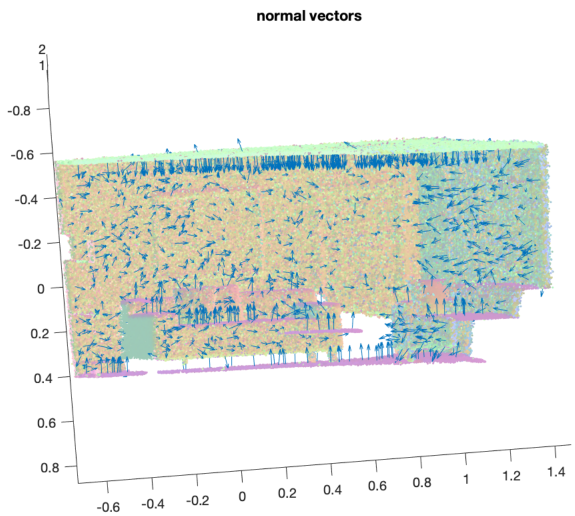


Figure 4: Point-Based Fusion - View 3

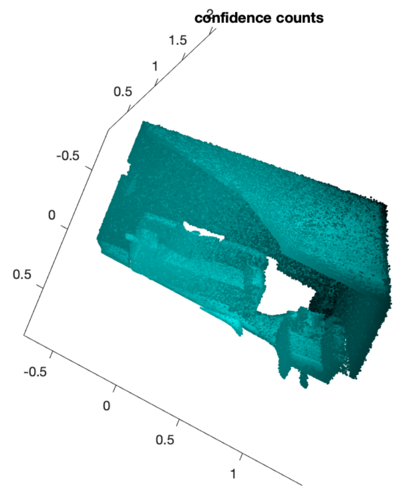
the better the model performs. The inliers measure all the matched point pairs such that in every iteration, the new points match the reference points. The higher the number of inliers, the better the model performs. This results in a better registration strategy and helps to remove outliers in the system more effectively.

The compression ratio can be computed as $\frac{\#\text{ points in output fusion map}}{\#\text{ points in entire input sequence}}$. This also means, if we can represent the scene geometry with a lesser number yet more accurate points, then the model performs better, translating to a lower compression ratio meaning a better model performance.

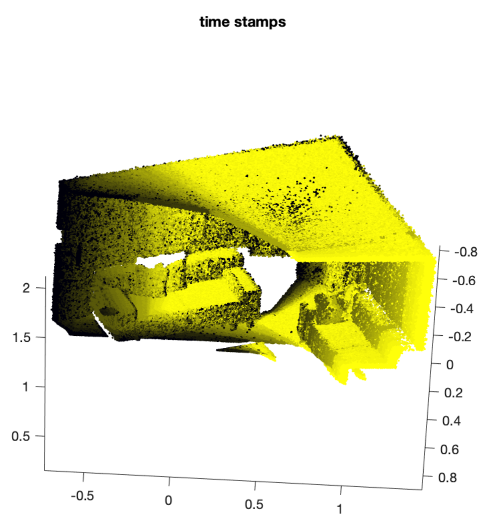
(c) The results of setting `is_eval = 1` are given below in Figure 5. The file `summary.txt` contains the output of the system in it.



(a) Surface Normal Vectors



(b) Confidence Counts



(c) Time Stamps

Figure 5: Results of Evaluation Maps

References

[1] Keller, Maik, et al. Real-time 3D reconstruction in dynamic scenes using point-based fusion. International Conference on 3D vision (3DV), 2013.

[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.

[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.