

# Pose Estimation using ICP

Nischal K N - nischal@seas.upenn.edu

**Abstract**—This project aims at determining the pose of an object using the 3D point cloud obtained from a depth camera. The pose is determined with respect to a 3D model of the object using iterative closest point(ICP). Two techniques are tried and compared here. Point to Point ICP and Point to Plane ICP are evaluated and the results obtained from both the methods are tabulated.

## I. INTRODUCTION

Iterative closest point is an algorithm to register the difference between two point clouds. Here in this project it is used to determine the pose of a object with respect to its 3D model. The 3D model is used as a frame of reference. From this it is possible to determine the pose of the camera and trace the movement of camera wrt to the model. The performance of two ICP algorithm, namely point to point and point to plane is evaluated and the results are presented in Section VII. The dataset used is explained in Section II. First the object of interest is extracted from the point cloud using RANSAC and some filtering as explained in Section III. The implementation of the two ICP are presented in Section IV and V.

## II. DATASET

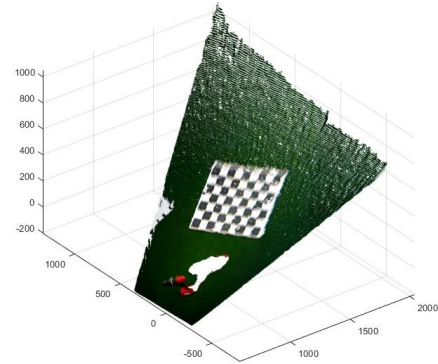
The dataset used for the project consists of depth and RGB images of 2 objects, a drill and an liquid container. It consists of a series of 447 depth and RGB images as the camera moves around the object of interest. Also a 3D model of the objects are used as the reference frame shown in Fig. 1c. The vicon data provided for the camera pose is not used. Fig 1 shows a sample RGB image and the corresponding point cloud.

## III. GROUND REMOVAL AND FILTERING

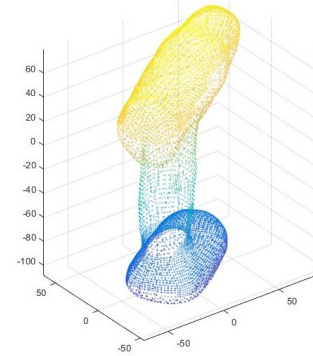
The first step is to extract the object from the 3D point cloud. This is performed in three steps. First a threshold is set on the maximum depth of the point cloud since the object is present close to the camera. Then RANSAC is applied to detect the ground plane with a very small threshold of 15mm and for 100 iterations. This removes most of the ground plane but leaves behind some residue that are removed with filtering. The output after RANSAC is shown in Fig. 2b with removed ground plane in red and object points in green. To remove this residue, the median of the point cloud is calculated and any points within 120mm of the median are retained. This value is choosen because the object has an approximate dimension of  $160 \times 120 \times 200mm$ . Any points that are further than 120mm is removed. The output is shown in Fig. 2c.



(a) RGB image



(b) Point cloud of corresponding depth image



(c) Point cloud of 3D model

Fig. 1: Pose initialization by removing translation

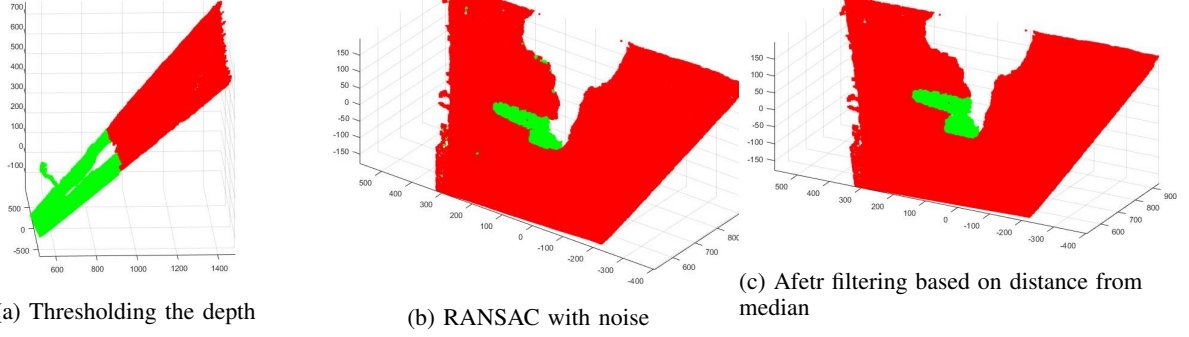


Fig. 2: Ground removal process. Green Pixels are retained and Red pixels eliminated at each step

#### IV. POINT TO POINT ICP

This technique follow the procedure described in [1]. It uses SVD to determine the rotation matrix. The algorithm is as follows

- 1) Determine the corresponding points of the model to each point of our point cloud using knnsearch.
- 2) Determine  $p$  and  $p'$  given as

$$p = \frac{1}{N} \sum_{i=1}^N p_i$$

$$p' = \frac{1}{N} \sum_{i=1}^N p'_i$$

where  $p_i$  and  $p'_i$  are the points in the model and depth point clouds.

- 3) Determine  $q$  and  $q'$  using

$$q_i = p_i - p$$

$$q'_i = p'_i - p'$$

- 4) Calculate the 3x3 matrix  $H$  and find its SVD

$$H = \sum_{i=1}^N q_i q_i^t$$

$$H = U \Lambda V^t$$

- 5) Finally calculate the Rotation and translation matrix using

$$R = V U^t$$

$$P = p' - R p$$

This procedure is repeated iteratively by using the determined  $R$  and  $T$  to initial the pose of the point cloud for the next iteration until the value of  $R$  and  $T$  stabilizes.

#### V. POINT TO PLANE ICP

Another version of ICP tested was point to plane[2]. Here the surface normals of the models are computed which are then compared with the point cloud to determine  $R$  and  $T$ .

- 1) Calculate the normals for all points of the model point cloud.
- 2) Do a KNN correspondence search to determine the closest model normals for the object point cloud.
- 3) Calculate the  $b$  matrix and  $A$  matrix as follows

$$b = \begin{bmatrix} n_{1x}d_{1x} + n_{1y}d_{1y} + n_{1z}d_{1z} \dots \\ \dots - n_{1x}s_{1x} - n_{1y}s_{1y} - n_{1z}s_{1z} \\ n_{2x}d_{2x} + n_{2y}d_{2y} + n_{2z}d_{2z} \dots \\ \dots - n_{2x}s_{2x} - n_{2y}s_{2y} - n_{2z}s_{2z} \\ \vdots \\ n_{Nx}d_{Nx} + n_{Ny}d_{Ny} + n_{Nz}d_{Nz} \dots \\ \dots - n_{Nx}s_{Nx} - n_{Ny}s_{Ny} - n_{Nz}s_{Nz} \end{bmatrix}$$

Where  $d$  is the points in model point cloud,  $s$  are the points in the object point cloud and  $n$  are the normals

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & n_{1x} & n_{1y} & n_{1z} \\ a_{21} & a_{22} & a_{23} & n_{2x} & n_{2y} & n_{2z} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{N1} & a_{N2} & a_{N3} & n_{Nx} & n_{Ny} & n_{Nz} \end{bmatrix}$$

where

$$a_{i1} = n_{iz}s_{iy} - n_{iy}s_{iz}$$

$$a_{i2} = n_{ix}s_{iz} - n_{iz}s_{ix}$$

$$a_{i3} = n_{iy}s_{ix} - n_{ix}s_{iy}$$

- 4)  $x$  consisting of  $R$  and  $T$  is calculated using

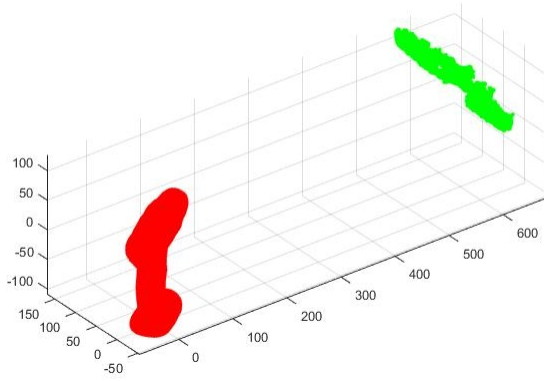
$$x = A^{-1}b$$

where

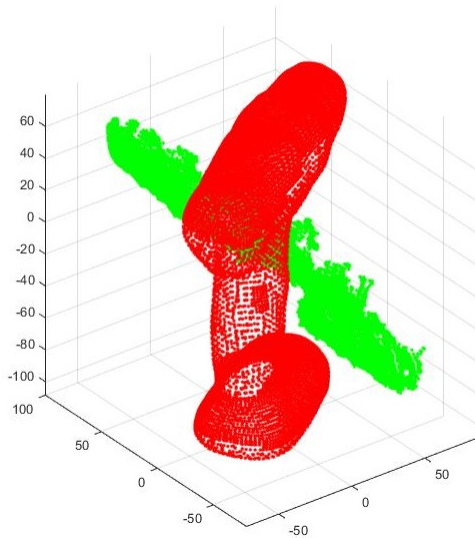
$$x = [\alpha \quad \beta \quad \gamma \quad t_x \quad t_y \quad t_z]$$

The  $R$  and  $T$  matrix are obtained from  $x$

This procedure is repeated iteratively by using the determined  $R$  and  $T$  to initial the pose of the point cloud for the next iteration until the value of  $R$  and  $T$  stabilizes.



(a) Initial pose with large translation



(b) Large pose corrected by mean centering

Fig. 3: Pose initialization by removing translation

## VI. INITIALIZATION OF ICP POSE

ICP algorithm is a local algorithm. Hence an approximate initial pose must be defined so that it converges well. In order to remove the large translation, the object is mean centered. This change is shown in Fig. 3b. Then two approaches were experimented to define the initial pose, coarse search and initializing with pose of previous frame.

### Coarse Search

The initial pose of the object was defined to be from a set of pitch and yaw angles 0 to 180 at increments of 30 degree. For each of the combinations of pitch and yaw, ICP was run for 50 iterations and the error was recorded. The error was measured as the sum of distance between the correspondence points obtained by knn search. The pair of pitch and yaw angles that had the minimum error was chosen as the initial pose for the ICP for that frame. However this method was too slow due to the number of iteration of ICP every frame had to go through.

### Initializing with pose of previous frame

This method was used to overcome the slow speed of coarse search. Instead of searching through the entire space of pitch and yaw angles, the pose obtained from the previous frame was used as the initial pose for the current frame and then the incremental pose change was computed. This made is sufficiently quicker. The initial pose was set manually

## VII. EXPERIMENTS AND RESULTS

For both the datasets, both the above mentioned ICP was run by initializing the pose with the pose of the previous frame. The results of some of the frames are tabulated in Table I and II. The number of iterations required to converge and the time taken are also specified. It is seen that point to plane ICP performs much faster than point to point ICP and also was more robust in converging. Additionally a link to GIF image of the process of convergence for each of these are provided.

## VIII. FUTURE WORK

- The R and T values calculated for each frame can be used to find the camera pose/camera motion around the object.
- Go ICP can be used to find the global minima.

## REFERENCES

- [1] Arun, K. Somani, Thomas S. Huang, and Steven D. Blostein. "Least-squares fitting of two 3-D point sets." Pattern Analysis and Machine Intelligence, IEEE Transactions on 5 (1987): 698-700.
- [2] Low, Kok-Lim. "Linear least-squares optimization for point-to-plane icp surface registration." Chapel Hill, University of North Carolina 4 (2004).

TABLE I: Comparison of point to point and point to plane ICP for drill

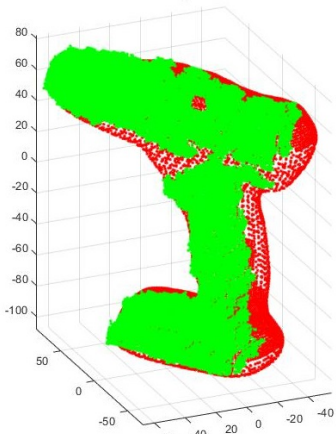
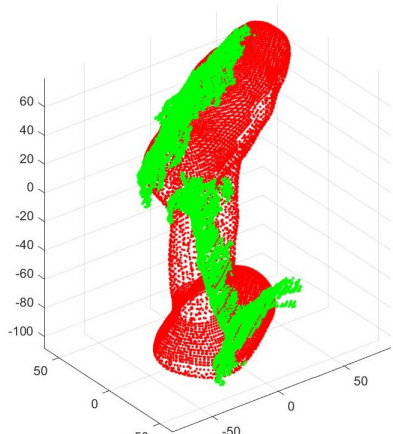
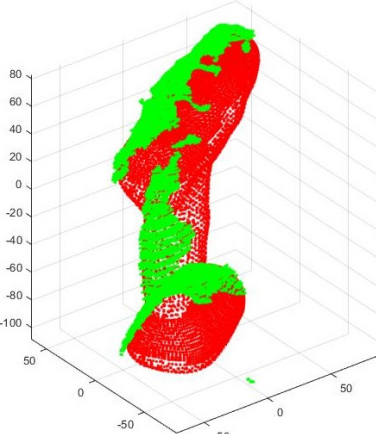
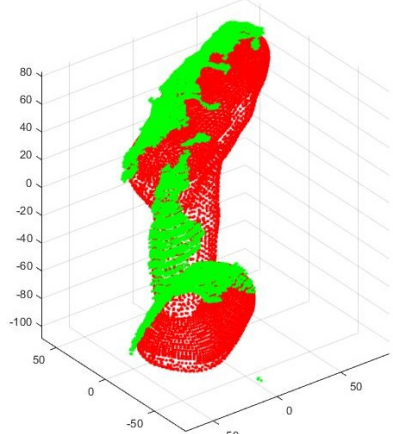
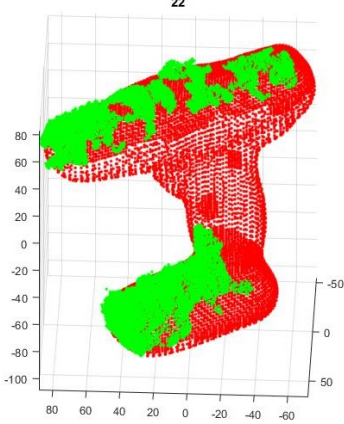
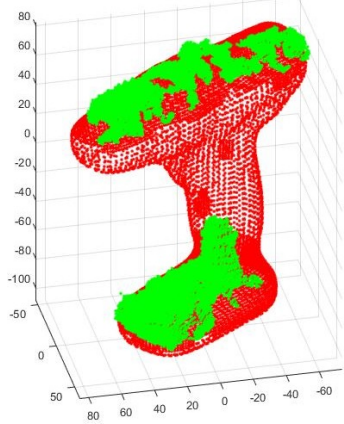
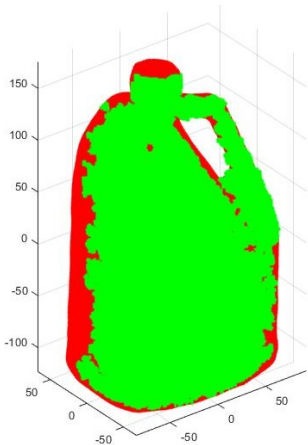
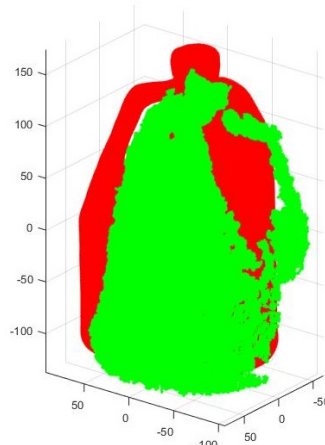
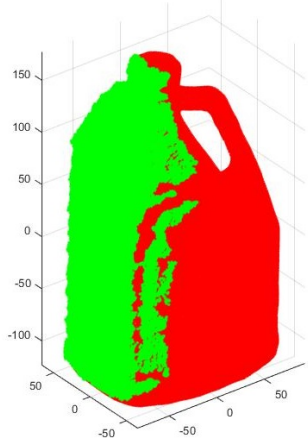
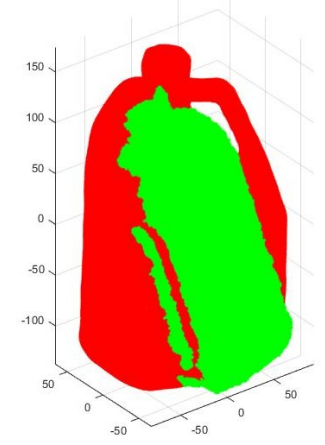
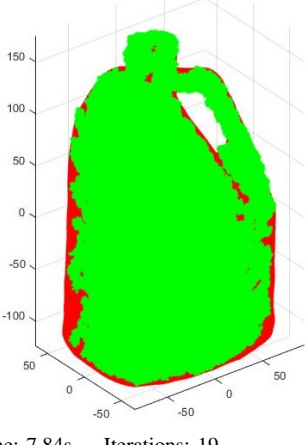
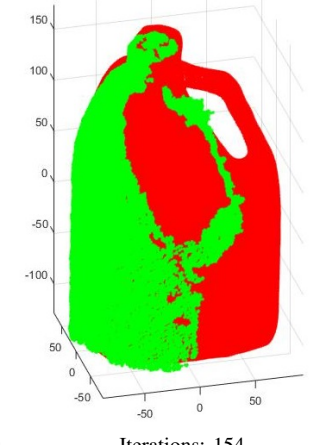
Frame Number	Point to Pplane ICP	Point to Point ICP
1	<p>9</p>  <p>Time: 2.12s Iterations: 9 <a href="#">GIF(click)</a></p>	<p>66</p>  <p>Time: 11.18s Iterations: 66 <a href="#">GIF(click)</a></p>
200	<p>11</p>  <p>Time: 2.30s Iterations: 11</p>	<p>69</p>  <p>Time: 10.92s Iterations: 69</p>
400	<p>22</p>  <p>Time: 3.60s Iterations: 22</p>	<p>63</p>  <p>Time: 9.57s Iterations: 63</p>

TABLE II: Comparison of point to point and point to plane ICP for liquid container

Frame Number	Point to Point ICP		Point to Plane ICP	
10	<div>15</div> 		<div>100</div> 	
	Time: 6.42s	Iterations: 15 <a href="#">GIF(click)</a>	Time: 44.72s	Iterations: 100 (Did not converge) <a href="#">GIF(click)</a>
210	<div>15</div> 		<div>136</div> 	
	Time: 6.80s	Iterations: 15	Time: 74.50s	Iterations: 136
410	<div>19</div> 		<div>154</div> 	
	Time: 7.84s	Iterations: 19	Time: 62.49s	Iterations: 154