B.TECH. PROJECT REPORT

On

Robust Technique to Align 3D Objects

BY

Kritik Sharma (160001031)

Rahul Kumar Keshri (140001023)

DISCIPLINE OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY INDORE

December 2019

Robust Technique to Align 3D Objects

A Project Report

Submitted in partial fulfillment of the requirements for the award of the degree

of
BACHELOR OF TECHNOLOGY
in

COMPUTER SCIENCE AND ENGINEERING

Submitted by:
Kritik Sharma (160001031)
Rahul Kumar Keshri (140001023)

Guided by:
Dr. Surya Prakash
Associate Professor, Discipline of Computer Science and Engineering, IIT Indore

INDIAN INSTITUTE OF TECHNOLOGY INDORE

December 2019

CANDIDATE'S DECLARATION

We hereby declare that the project entitled **"Robust Technique to Align 3D Objects"** submitted in partial fulfillment for the award of the degree of Bachelor of Technology in **Computer Science and Engineering** completed under the supervision of **Dr. Surya Prakash, Associate Professor, CSE, IIT Indore** is an authentic work.

Further, I/we declare that I/we have not submitted this work for the award of any other degree elsewhere.

Kritik Sharma	Rahul Kumar Keshri
Date:	Date:

CERTIFICATE by BTP Guide

This is to certify that the thesis entitled **"Robust Technique to Align 3D Objects"** and submitted by Kritik Sharma, Roll No. 160001031 and Rahul Kumar Keshri, Roll No. 140001023, in partial fulfillment of the requirements for CS 493 B.Tech Project embodies the work done by them under my supervision. It is certified that the declaration made by the students is correct to the best of my knowledge.

Signature of the Supervisor

Dr. SURYA PRAKASH

Associate Professor

Discipline of Computer Science and Engineering

Indian Institute of Technology Indore

Date:

Acknowledgments

We would like to take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of our project.

We would like to express our sincere gratitude to **Dr. Surya Prakash**, our project supervisor, for his valuable suggestions and keen interest through out the progress of our project.

We would like to thank **Mr. G. Iyappan** for his valuable suggestions throughout the project.

Last but not the least, we would like to thank our family members, classmates and other students of IIT Indore for their support.

With Sincere Regards,

Kritik Sharma

Rahul Kumar Keshri

Discipline of Computer Science & Engineering

Indian Institute of Technology Indore

Abstract

In literature, the Iterative Closest Point (ICP) algorithm is used to align 3D objects on the basis of their coordinates and uses the Euclidean distance between them as an error of registration, which is inaccurate most of the time. Nevertheless, after some iterations, it performs much better in reducing the error as much as possible as the iteration does. On the other hand, Point Feature Histogram (PFH) is used to align 3D objects by computing their PFH at each point and then using them to locate the corresponding points in the other point cloud. Since ICP uses distances to compute the corresponding points, it is strongly affected by how two 3D objects are positioned in the coordinate system at the beginning and as soon as the transformation is applied. Each coordinate shifts and now we have to measure again the distances between the two point clouds to find a better alignment than before.

In a quest for further improvement, we used combination of PFH, RANSAC and ICP to align 3D objects more efficiently with the time taken to make it more feasible to compare objects with a large number of points. This is achieved by using a point feature histogram and using a pre-determined threshold to consider two points as corresponding points. Further, RANSAC algorithm is used to find inliers of the corresponding points to compute a transformation matrix which is used to register both point clouds. In addition, ICP is used to finely align both clouds for better time efficiency and less error.

This report shows the use of combinations of PFH, RANSAC and ICP to reduce the time taken to align 3D objects with comparable or better performance compared to ICP.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter highlights the background and motivation of this project The problem statement of the project has been identified and the significance of the results has also been clearly explained.

1.1 Background

As the world is in awe, just by looking at Sophia an Artificial Intelligence (AI) robot [1]. The way robots look at this environment and this whole world is very different from humans. As humans, we see everything around us in the form of images. AI Robots can't do the same thing as camera angles, sensors and lights can lead to overexposure, and underexposure of objects which may lead to a different perception of 2D images than is expected. That is why it is much better to use sensors and scanners to see everything around them in the form of 3D images.

Nowadays, it is fairly much easier to steal username or PIN of someone for access to information that is too important for the organization It is not safe to use a password or a PIN to secure them. Therefore, using a biometric to identify an actual user is a much better and more reliable alternative than the former. There's a lot of research going on in recent decades to find a way to distinguish two 3D scans so that we can differentiate the Genuine user from the Imposter.

In this report, we would like to address how our technique works relative to how PFH and RANSAC and ICP work. The aim behind this project is to build a technique for aligning 3D objects to make the registration of two 3D scans more accurate and time-efficient.

In the last decades, a group of techniques proposed in the literature for 3D recognition and registration, such as ICP [2], Point Feature Histogram (PFH) [3], Fast Point Feature Histogram (FPFH) [4], shot descriptor [5], spin images [2], Normal Aligned Radial Features (NARF) [6], Normal Distribution Tranforms (NDT) [7], RANSAC [8]. They all have their own disadvantages and advantages. So instead of using them as a single method, we use a combination of selected techniques *viz.* PFH, RANSAC and ICP to improve efficiency during the registration of two 3D scans.

1.2 Objectives

As described in the above discussion, the goal is to align two 3D point Clouds more effectively with less time; and also to make them scalable for handling Big Data. We propose a technique that delivers comparable or better results than ICP, in terms of computational cost and registration error.

Chapter 2

Preliminaries

This chapter discusses the different concepts used in the course of this project This chapter starts with a discussion on PFH and a summary of the different methods for aligning two 3D point clouds. The end of this chapter consists of a review on ICP, its features and advantages.

2.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) uses orthogonal transformation to obtain a set of correlated variables with the help of variables that may be correlated with each other. PCA was invented by Karl Pearson [9], drawing inspiration from the principal axis theorem found in mechanics, then developed independently and named it by Harold Hotelling [10].

PCA is usually used as a tool for exploratory data analysis or for the generation of predictive models. PCA is the least complicated of all real multivariate analyzes dependent on the eigenvector. PCA can also be used to estimate the surface normal of the point cloud with the help of its neighbors.

$$C = \frac{1}{k}\sum_{i=1}^{k}(a_i - \overline{a})(a_i - \overline{a})^T$$

Here C is a covariance matrix, a_i are points in the data set and $\overline{a}$ is the centroid of points in data set, k is the number of points. Compute eigenvalues of the obtained covariance matrix using

$$|A - \lambda I| = 0$$

Using the calculated eigenvalues, eigenvectors can be obtained as described below

$$C.\overline{v_j} = \lambda_j.\overline{v_j} \ \ , j \in \{0, 1, 2\}$$

Here, eigenvector ($\overline{v_j}$) is corresponding eigenvector of eigenvalue ($\overline{\lambda_j}$).

2.2 Point Feature Histogram

Point Feature Histogram is used to identify point-to-point correspondence between two 3D point clouds and then use them to compute a transformation matrix. Further, the obtained matrix is used to align two point clouds to see how similar or different the point clouds are. Since most methods use surface normals and curvature estimates [11], etc., they do not fully describe the local surroundings of the k-neigbourhood, and most of the time these are singular values. Ideally, if we have information's like whether it's a point on edge or plane or sphere and so on. Using these features and some constraints, help us to increase the likelihood of having much better correspondence. In order to have this point information, it is much better to describe a point using histogram of values [12], [13] which contains more detail of the geometric properties of its neighborhood and presents an invariant multi-feature which doesn't depend on its orientation.

2.2.1 Computation of Point Feature Histogram

Steps to calculate PFH for a point is:-

- First, we calculate the normal of each and every point whose normal is not available with the help of its k-nearest neighbours and then using Principal Component Analysis to calculate it's normal.

- Once all of the normals are obtained we orient them according to viewpoint.

- Then, after re-orienting them, to calculate the point histogram, we first find the k-nearest neighbors within a radius R. Further, pair all neighbors with the query point to calculate the

$$< \alpha, \phi, \theta, d >$$

for each pair, from which one point is considered to be the source and the other is considered to be the target point.

- Source Point and target point is identified by

$$\langle\ n_i, p_i - p_j \rangle \leq \langle n_j, p_j - p_i \rangle$$

then

$$p_s = p_i, p_t = p_j$$

otherwise

$$p_s = p_j, p_t = p_i$$

Here, p_i and p_j are the two points for which, we need to calculate α, ϕ, θ, d, and p_s and p_t denotes which one is source and target respectively.

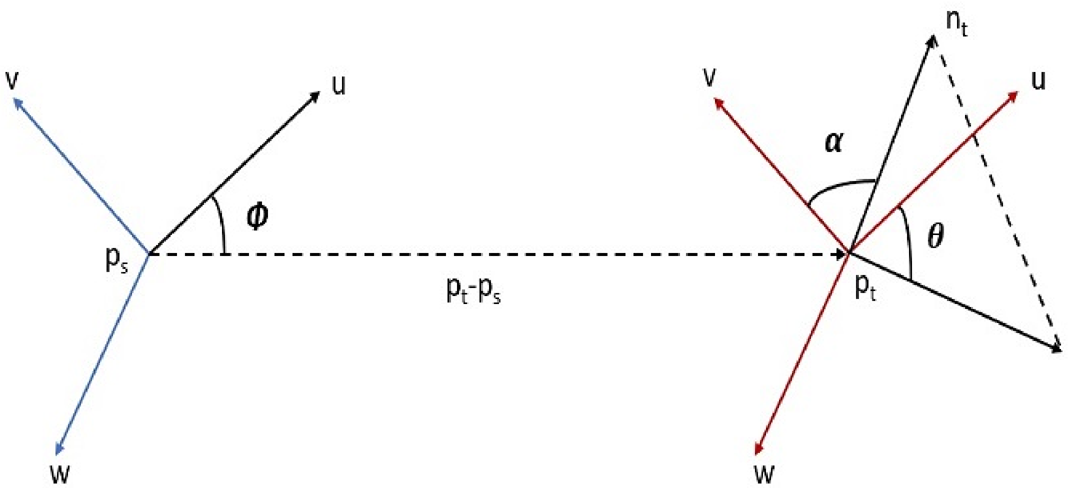

Figure 2.1: Darboux Frame

- After we have source and target point we will describe Darboux Frame (as shown in Figure 2.1) considering source point as origin as:

$$u = n_s,\ v = (p_t - p_s) \times u,\ w = u \times v$$

now,using UVW frame to calculate α, θ, ϕ and d as described below

$$\alpha = \langle v, n_t \rangle$$
$$d = ||p_t - p_s||_2$$
$$\phi = \langle v, p_t - p_s \rangle / d$$
$$\theta = atan(\langle w, n_t \rangle, \langle u, n_t \rangle)$$

- The set of quadruplets obtained for the query point, with the help of all pairs in the neighbourhood of the query point within the radius R, will then be binned to a histogram to obtain the PFH for the query point.

- The binning method divides the range of feature values into k subdivisions, and then counts the occurrence of every feature value in each interval.

2.2.2 Applications of Point Feature Histogram

Point Feature Histogram can be used in many ways:

- In finding salient points.

- To describe salient points.

- To find correspondence points.

2.3 Random Sample Consensus (RANSAC)

Random Sample Consensus (RANSAC) [8] algorithm is a general parameter approach designed to separate inliers and outliers from a large amount of input data. RANSAC is a re-sampling approach that generates candidate responses by using the minimum quantity of observations required to estimate the underlying model parameters. Unlike conventional sampling methods, this approach uses as much data as possible to obtain an initial response and then continues with the elimination of outliers, RANSAC makes use of the smallest set feasible and proceeds to extend this set with consistent data points [8].

2.3.1 Algorithm

Basic algorithm of Random Sample Consensus (RANSAC) is as follows

1. Out of all points in the input data, randomly choose some number of points (more than bare minimum) to calculate the model parameters.
2. Predetermine out of how many points from the set of all points in the point cloud containing inliers as well as outliers fit within an already considered tolerance ϵ.
3. If ratio of the number of inliers over the sum of number points in inliers and outliers exceeds a threshold τ , re-estimate the model with the help of identified inliers and terminate if the required conditions are met.
4. Otherwise, if number of iterations till now is less than N, then repeat steps 1 to 3 again

Here N denotes number of iterations, and is taken large enough to make sure that probability prob (most of the time it's 0.99), such that out of indefinite samples at least one will be without an outlier. Let's say probability to select a data point as an inlier is u and as an outlier is v=1-u, then the probability density of observing an outlier. After iterating the whole procedure N no. of times will be

$$1 - prob = (1 - u^m)^N$$

$$N = \frac{log(1 - prob)}{log(1 - (1 - v)^m)}$$

2.4 Iterative Closest Point (ICP)

The goal of the Iterative Closest Point (ICP) algorithm[14] is to find a transformation between two point clouds, considering one as a reference point cloud and another as a moving point cloud, by minimizing the square error between the corresponding entities. Iterative Closest Point (ICP) is also used to reconstruct 2D or 3D surfaces from various scans, to improve route planning, to localize robots, etc. The biggest drawback in the ICP is that it needs an initial alignment, otherwise it get stuck in local minimum and the alignment become meaningless. ICP converges to the nearest local minimum of the mean square distance, monotonically. In Early iterations, convergence in faster comparatively to the later iterations.

2.4.1 Algorithm

Steps to compute ICP is as follows

1. Each and every point in the fixed point cloud is matched with the closest point(in terms of Euclidean distance (used in classic ICP) in the moving point cloud.

   $$d(point1, point2) = \sqrt{((a_1 - a_2)^2 + (b_1 - b_2)^2 + (c_1 - c_2)^2)}$$

   Considering point point1=(a1,b1,c1) and point2=(a2,b2,c2)

2. Now estimating translation and rotation matrix with the help of RMS point to plane or point euclidean distance metric minimization which will align fixed point cloud in step 2, this step may optionally use weighting points for calculating alignment.

   $$f(R, T) = \frac{1}{N_d} min \sum_{i=1}^{N_d} ||p_i - R(d_i) - T||^2$$

   Here, N_d is size of moving point cloud, p_i is each point of fixed point cloud, d_i is each point of moving point cloud.

3. Transform the source point cloud after obtaining source point cloud.

4. Iterate step 1,2,3 until n-iterations or until achieving a local minima.

Chapter 3

Proposed Technique

In this chapter, we discuss the design and implementation of the robust techniques to align 3D objects based on Point Feature Histogram (PFH), Random Sample Consensus (RANSAC) and Iterative Closest Point (ICP).

3.1 Estimation of Normals

First step to calculate point feature histogram is to calculate normals at each point in the point cloud.
Algorithm to calculate normal at each and every point in the point feature histogram is discussed below:-

1. Do k-NN search on the basis of Euclidean distance to find 6 nearest neighbors at each point

2. To calculate the approximation of surface normal at a given point, first look for it's nearest neighbours, then calculate their co-variance matrix and then evaluate eigenvalues and eigenvectors to approximate the surface normal at that point. For each point a_i in the point cloud, calculating centroid of the query point consider it as p and considering a_i as it's neighbours.

$$\overline{p} = \sum_{i=1}^{k} a_i$$

3. Now calculate co-variance matrix by

$$C = \frac{1}{k}\sum_{i=1}^{k}(p_i - \overline{p})(p_i - \overline{p})^T$$

4. After getting the co-variance matrix, calculate eigenvalues and eigenvectors.

$$C.\overline{v_j} = \lambda_j.\overline{v_j}, j \in \{0, 1, 2\}$$

   Here, eigenvector $(\overline{v_j})$ is corresponding eigenvector of eigenvalue $(\overline{\lambda_j})$.

5. $\overline{v_0}$ is the eigenvector associated with the smallest eigenvalue and it is our estimated normal vector at that point.

6. Evaluate Step 2-6 for each and every point.

7. Now align point (p_i) with normal (N_i) according to viewpoint (let it be V) by:

$$\langle (p_i - V), N_i \rangle > 0$$

   then $N_i = -N_i$

8. Repeat Step 7 for all points

3.2 Computation of PFH using PCL Library

The PCL library contains off-shelf implementation of numerous algorithms such as feature estimation algorithms, model fitting, filtering, and registration for 2D and 3D images.

Point Cloud Library supports PCD format. We converted our point cloud to PCD format in pre-processing step to obtain normals for each pont clouds in the database.

To obtain a Point Feature Histogram at each point in the point cloud, the PFHEstimate requires input of point cloud in PCD format and it's corresponding normals in PCD format.

3.3 Obtaining Correspondance Points

3.3.1 Point Feature Histogram

Point Feature Histogram is one of the most efficient volume based descriptor. Let's describe two point cloud, with the help of point feature histogram instead of coordinates and normals, then look for corresponding points between them. The implementation in MATLAB has the following parameters:

1. With the help of k-NN search, find 2 nearest Point Feature Histogram in other point cloud using Euclidean distance.

2. If nearest neighbour is at a distance d_1 and second nearest neighbour is at a distance d_2, then

$$\frac{d_1}{d_2} < \Upsilon$$

   considering Υ, if a Point Feature Histogram (PFH) satisfies the criteria then it is the point at which the minimum distance d_1 was found, is the corresponding point of query point in the other point cloud.

3. Repeat Step 2 for every point in the point cloud.

After the whole process, we followed above, with the help of Point Feature Histogram, we have point correspondences between two point clouds.

3.3.2 Random Sample Consensus (RANSAC)

RANSAC is interpreted as an outlier detection method[8]. It is a non-determininstic algorithm which results into a reasonable result only with help of more number of iterations, as it results with a certain probability.

We will remove outliers from the corresponding matches to get more improved matches among them.The implementation in MATLAB has the following parameters:

- p: Inliers percentage (default 0.25)
- sigma: noise std (default: 20)
- epsilon: False Alarm Rate (default=10^{-6})

- P_inlier: Chi squared probability threshold (default=0.9999)
- min iterations: minimum number of iterations (default = 0)
- mode: we used RANSAC (default='MSAC')

After this approach, we get a collection of corresponding points after removing the outliers to have a more efficient transformation matrix to register point clouds.

3.4 Iterative Closest Point

This technique is used to register two pointclouds to find the alignment in terms of registration error between them. Steps of the modified algorithm:

1. After using RANSAC we have a subset of points in both the point clouds
2. We are applying ICP over it to calculate an initial transformation matrix R and T, Here R being rotation matrix and T being translation matrix.
3. Then applying the transformation matrix on the whole moving point cloud to obtain a new transformed point cloud
4. Then again applying ICP on the whole transformed point cloud and fixed point cloud.
5. Then compute the final transformed point cloud and then check the error between the two point clouds after the transformation with the help of

$$error = \sum_{n=1}^{k} |p_i - q_i(match)|$$

, here *match* denotes corresponding match of p_i in q_i.

During this alignment procedure we used a volume-based descriptor PFH to calculate a volume-based descriptor instead of a distance which is too dependent on the relative position of two point clouds. Further, after obtaining the corresponding point clouds, we used RANSAC to distinguish inliers and outliers to obtain a perfect corresponding matches.

Further, applying ICP over those corresponding coordinates, we got transformation matrix and then we applied that transformation matrix on the whole point cloud to obtain aligned point clouds. Then we apply ICP on the transformed point cloud to obtain a finely aligned point clouds and error between them. In Figure 3.1, PC 1 and PC 2 are two point clouds. After applying normal computation on both of them, to obtain PC 1+Normal and PC 2+Normal then,after applying PFHEstimate on both of them, to obtain PC 1+Normal+PFH and PC 2+Normal+PFH then after applying technique to find corresponding points, we have PC 1' and PC 2' these point clouds are corresponding points in the respective point clouds of PC 1 and PC 2, which are matching the criteria of corresponding points of PFH and then applying RANSAC on PC 1' and PC 2' to seperate inliers and outliers, and PC 1" and PC 2" containing corresponding points having efficient matches, now using PC 1" and PC 2" and apply ICP over them to get a transformation matrix, now applying transformation matrix on whole PC 1 to get a transformed point cloud tr. PC 1 and then applying ICP over Tr. PC 1 and PC 2 to get registration error. Flowchart of proposed technique is:

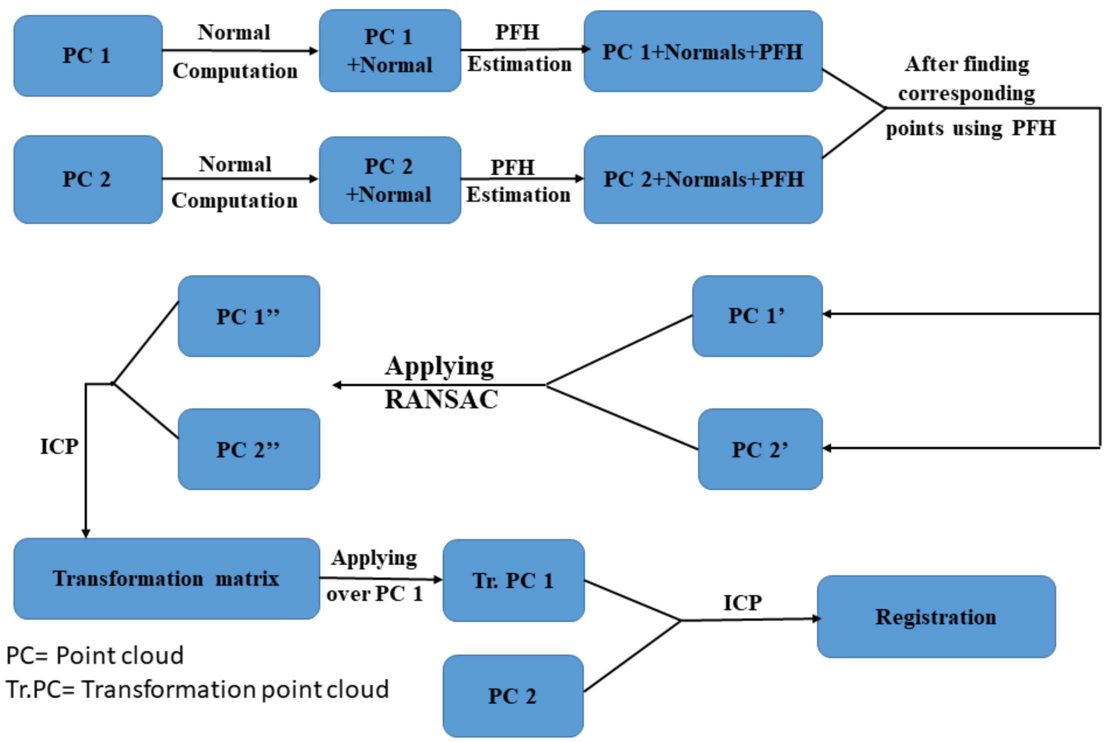

Figure 3.1: Flowchart of the proposed technique

Chapter 4

Experimental Analysis

This chapter presents the results obtained by implementing the Alignment Procedure discussed in Chapter 3 comparatively to Alginment using only Iterative Closest Point(ICP). A description of the datasets used for experimentation are:

UND Dataset

The University of Notre Dame (UND) J2 ear data- base is the largest openly accessible ear database with 415 subjects and 1800 samples.

IITI Dataset

This database is collected by IIT Indore and this database consists of persons with a range of 18 to 60. It consists of 3 left ear and 3 right ear images.These images have some noises,therefore we need to pre-process these images before using them for any recognition purposes.

4.1 Computation of Optimal Radius for PFH

This section has the experimental data of how the value of radius affects, such that the set of PFH for a 3D object is as distinct as possible.To find the optimal radius, compare ever point's PFH to every other point's PFH and then check how much percentage of those values are distinct.

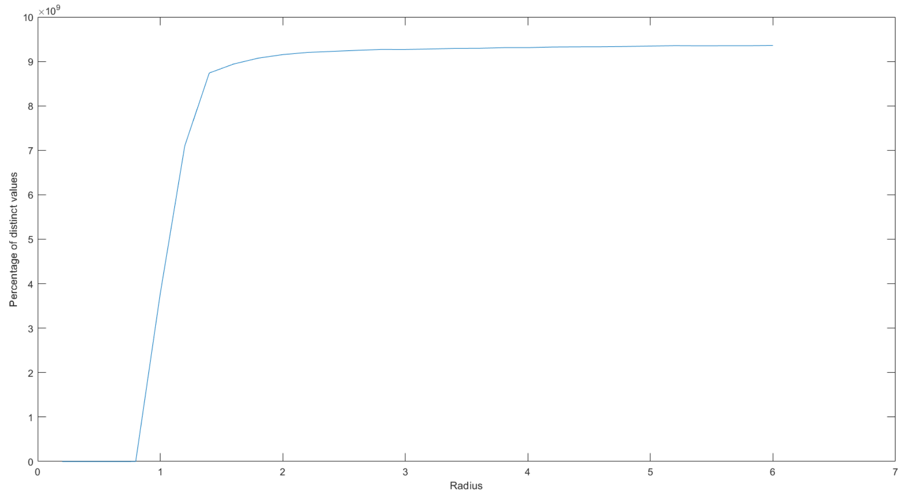

Figure 4.1: Percentage of distinct histograms vs. Radius

4.2 Alignment Accuracy

Here, we will discuss how efficiently, proposed technique can differentiINate between similar and different objects. In proposed technique, at first we are using PFH to find corresponding points between two point cloud, then we are seperating corresponding inlier points from outlier points using RANSAC (considering τ as 25% and ϵ as 0.9999) , then on set of inliers after using ICP (10 iterations) and then, applying the output transformation of the ICP on the moving point cloud, then using ICP (10 iterations) on transformed point cloud and fixed point cloud, then finally calculate registration error and time taken and then comparing it by using ICP (20 iterations) After using 5 Ear samples of all 50 humans, 5 ear samples of every human is compared to all other samples of same human and then we recorded data of both proposed technique and classic ICP.

Comparison of similar objects

This section will show the analysis of error between robust technique and ICP algorithm, when we are comparing similar objects. If both the Ear samples are of same human then ideally error should decrease, and after applying proposed technique instead of classic ICP.

Let's say error by ICP is $Eicp$ and error by proposed technique is Ero.

Algorithm to identify better,worse or comparable result is:

1. For Comparable

$$Ero - Eicp < \tau$$

2. For better

$$Ero - Eicp < 0$$

3. Otherwise worse

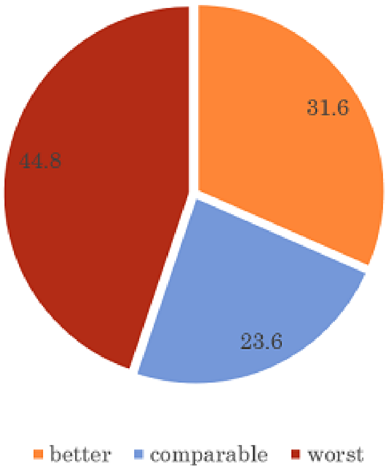

Figure 4.2: Error while comparing similar ear on UND dataset

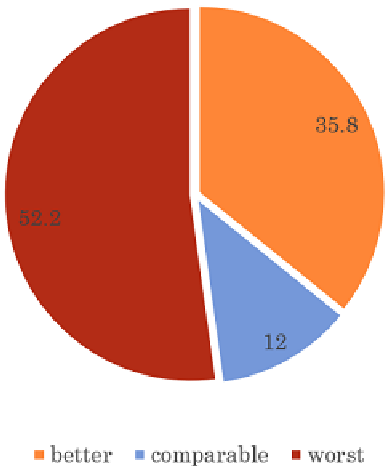

Figure 4.3: Error while comparing similar ear on IITI dataset

In Figures 4.2 and 4.3 represents the percentage of times registration error calculated from proposed technique (while comparing similar ear) was better using proposed technique in orange, percentage of time proposed technique and ICP (20 iterations) was comparable in blue and ICP (20 iterations) was better in red.

Comparison of different objects

This section will show the analysis of error between proposed technique and ICP algorithm when we are comparing similar objects. If both the Ear samples are of different human then ideally error should increase after applying proposed technique instead of classic ICP.
Let's say error by ICP is $Eicp$ and error by proposed technique is Ero. Algorithm to identify better,worse or comparable result is:

1. For Comparable
$$Eicp_i - Ero_i < \tau$$
2. For better
$$Ero_i - Eicp_i > 0$$
3. Otherwise worse

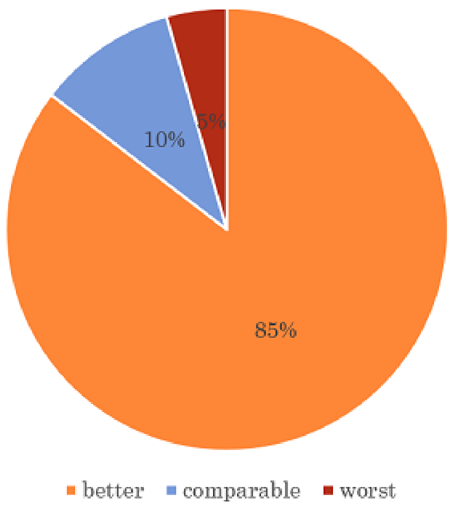

Figure 4.4: Error while comparing different ear on UND dataset

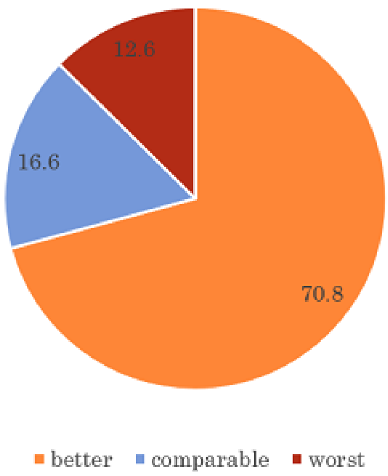

Figure 4.5: Error while comparing different ear on IITI dataset

In Figures 4.4 and 4.5 represents the percentage of times registration error calculated from proposed technique (while comparing different ear) was better using proposed technique in orange, percentage of time proposed technique and ICP (20 iterations) was comparable in blue and ICP (20 iterations) was better in red.

4.3 Computational Time

Here, we will discuss how time efficiently our method is comparatively to ICP. If we compare the time taken by proposed alignment technique comparatively to ICP (after 20 iterations). Majority of the times Proposed alignment technique is much more time efficient with better or comparative result than ICP.

Comparison of similar objects

In case of UND dataset, after doing 500 comparisions between similar ear, 500/500 times proposed technique is taking less time comparatively to ICP. In Figure 4.6, it is plot of time taken for comparison (in seconds) by proposed technique (showed in blue) and by ICP (showed in red) for UND dataset while comparing similar ear.

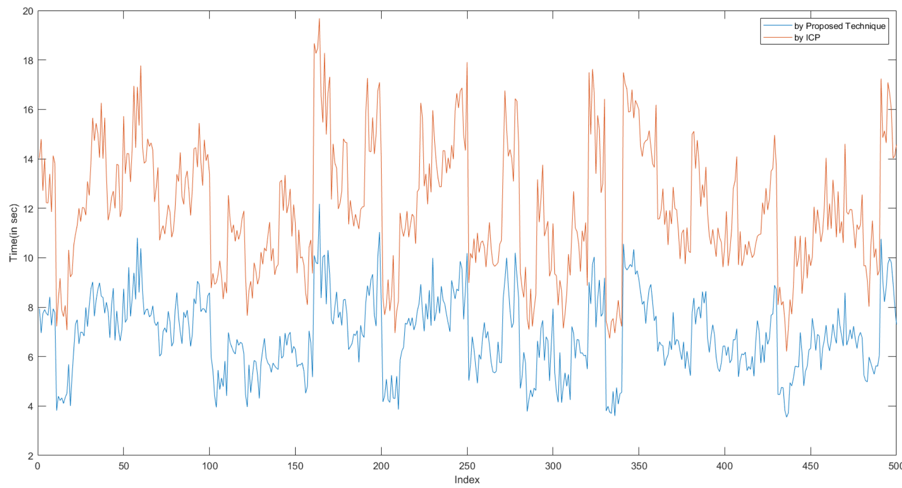

Figure 4.6: Time Comparison (ICP vs. Proposed Technique) on UND dataset (similar ear)

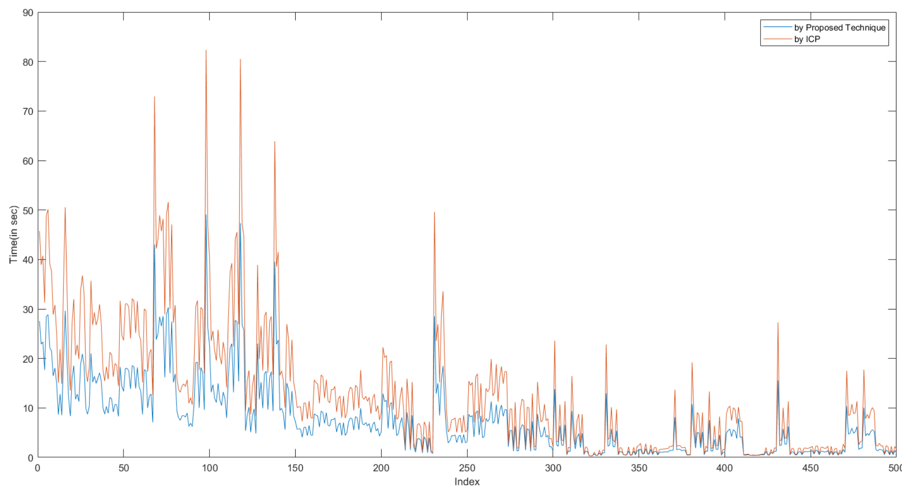

Figure 4.7: Time Comparison (ICP vs. Proposed Technique) on IITI dataset (similar ear)

In case of IITI dataset, comparisions between similar ear, 496/500 times proposed technique is taking less time comparatively to ICP.In Figure 4.7, it is plot of time taken for comparison (in seconds) by proposed technique (showed in blue) and by ICP (showed in red) for IITI dataset while comparing similar ear.

Comparison of different objects

In case of UND dataset, comparisions between different ear, 500/500 times proposed technique is taking less time comparatively to ICP.In Figure 4.8, it is plot

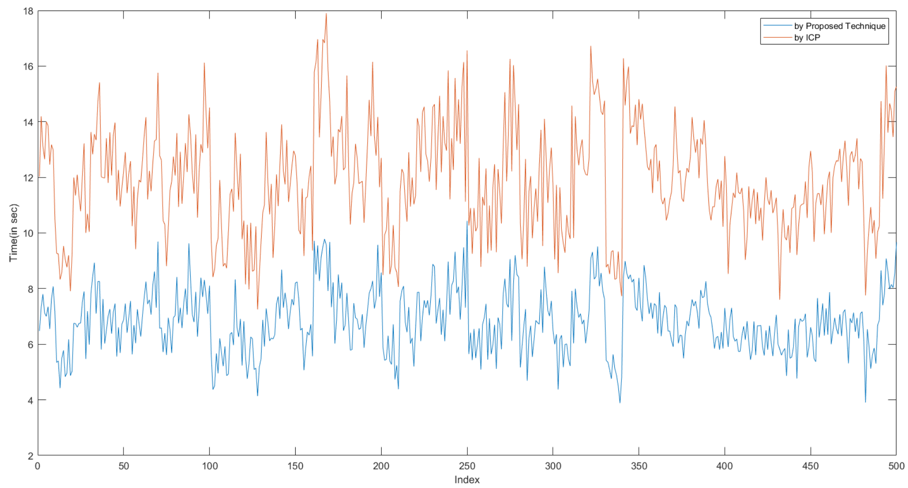

Figure 4.8: Time Comparison (ICP vs. Proposed Technique) on UND dataset (different ear)

of time taken for comparison (in seconds) by proposed technique (showed in blue) and by ICP (showed in red) for UND dataset while comparing different ear.

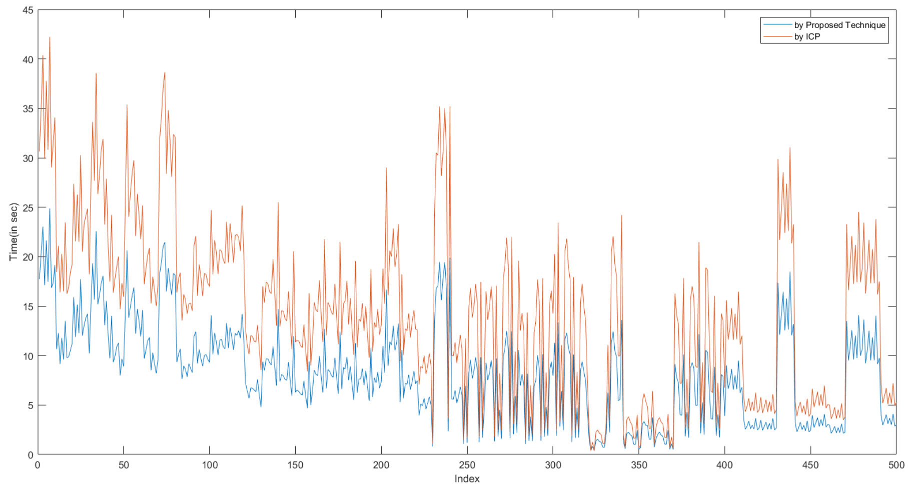

Figure 4.9: Time Comparison (ICP vs. Proposed Technique) on IITI dataset (different ear)

In the case of IITI dataset, comparisions between different ear, 498/500 times proposed technique is taking less time comparatively to ICP.In Figure 4.9, it is plot of time taken for comparison (in seconds) by proposed technique (showed in blue) and by ICP (showed in red) for UND dataset while comparing different ear.

Table 4.1: Time comparison (ICP vs. proposed technique) on IITI dataset (similar ear)

Technique	**average time (in sec)**	**total time (in sec)**
ICP	13.146	6573.084
Proposed Technique	7.599	3799.664

Table 4.2: Time comparison (ICP vs. proposed technique) on IITI dataset (different ear)

Technique	**average time (in sec)**	**total time (in sec)**
ICP	14.302	7150.77
Proposed Technique	8.121	4060.468

Table 4.3: Time comparison (ICP vs. proposed technique) on UND dataset (similar ear)

Technique	**average time (in sec)**	**total time (in sec)**
ICP	11.975	5987.581
Proposed Technique	6.877	3438.700

Table 4.4: Time comparison (ICP vs. proposed technique) on UND dataset (different ear)

Technique	**average time (in sec)**	**total time (in sec)**
ICP	11.905	5952.372
Proposed Technique	6.835	3417.617

In Tables 4.1 - 4.4 represent the time calculated while doing experimentation on UND and IITI data sets. Here, average time represents, the average time during 500 comparisons between similar or different ear (as mentioned in the table) and total time represents, the total time during 500 comparison between similar or different ear (as mentioned in the table).

Chapter 5

Conclusion and Future Scope

The classic Iterative Closest Point (ICP) algorithm uses point-to-point correspondence based on the Euclidean distance between their coordinates, which is highly dependent on how the cloud is aligned with each other. The main drawback of using ICP is that it gets stuck in its local minima. So, to overcome this, we propose a headstart to ICP so that it doesn't get trapped in its local minima and thus results in a better alignment. Also, instead of using the complete data, we used PFH to find the corresponding points and further used RANSAC to extract the perfect inliers from the corresponding points. Further, these corresponding points are used to compute the transformation matrix. Then the obtained transformation matrix is applied on the whole point cloud (moving), and after that we apply ICP on the whole fixed point cloud and transformed point cloud. The proposed technique takes less time to align 3D objects compared to ICP to align 3D objects most of the time, although the proposed algorithm provides comparable and better results most of the time compared to ICP. In future, we can use other ways to compare the error between two point feature histogram to find corresponding points. Therefore,leading to more efficient corresponding matches, therefore leading to better alignment head start then before, such that we will be having much lesser error then the proposed algorithm leading to better alignment of two 3D objects.

Bibliography

[1] Robin R. Murphy. *Introduction to AI Robotics*. MIT Press, Cambridge, MA, USA, 1st edition, 2000.

[2] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):433–449, May 1999.

[3] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz. Aligning point cloud views using persistent feature histograms. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3384–3391, Sep. 2008.

[4] R. B. Rusu, N. Blodow, and M. Beetz. Fast Point Feature Histograms (FPFH) for 3D registration. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 3212–3217, May 2009.

[5] Samuele Salti, Federico Tombari, and Luigi Di Stefano. Shot: Unique signatures of histograms for surface and texture description. *Computer Vision and Image Understanding*, 125:251 – 264, 2014.

[6] Bastian Steder, Bogdan Rusu, Kurt Konolige, and Wolfram Burgard. Narf: 3D range image features for object recognition. Jan 2010.

[7] P. Biber and W. Strasser. The normal distributions transform: a new approach to laser scan matching. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 2743–2748 vol.3, Oct 2003.

[8] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *ACM Commun.*, 24(6):381–395, June 1981.

[9] Karl Pearson F.R.S. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[10] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, 1933.

[11] Kwang-ho Bae and Derek Lichti. Automated registration of unorganised point clouds from terrestrial laser scanners. *International Archives of Photogrammetry and Remote Sensing*, 35, July 2004.

[12] Radu Bogdan Rusu, N. Blodow, Z. Marton, A. Soos, and M. Beetz. Towards 3D object maps for autonomous household robots. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3191–3198, Oct 2007.

[13] E. Wahl, U. Hillenbrand, and G. Hirzinger. Surflet-pair-relation histograms: a statistical 3D-shape representation for rapid classification. In *Proceedings of International Conference on 3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings.*, pages 474–481, Oct 2003.

[14] P. J. Besl and N. D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, Feb 1992.

Appendix A

Setup Guidelines

A.1 Setting up Environment

A.1.1 MATLAB & PCL

In order to get point cloud library and Matlab running on your systems, Matlab(Matlab 19) and PCL(v1.8) are required. This section provided instruction to install the same.

Installing PCL

1. Open terminal
2. Enter these commands in terminal

   - sudo add-apt-repository -y ppa:webupd8team/java
   - sudo apt update
   - sudo apt -y install oracle-java8-installer
   - sudo apt -y install g++ cmake cmake-gui doxygen mpi-default-dev openmpi-bin openmpi-common libusb-1.0-0-dev libqhull* libusb-dev libgtest-dev
   - sudo apt -y install git-core freeglut3-dev pkg-config build-essential libxmu-dev libxi-dev libphonon-dev libphonon-dev phonon-backend-gstreamer
   - sudo apt -y install phonon-backend-vlc graphviz mono-complete qt-sdk libflann-dev

- sudo apt -y install libflann1.8 libboost1.58-all-dev
- cd /Downloads
- wget `http://launchpadlibrarian.net/209530212/libeigen3-dev_3.2.5-4_all.deb`
- sudo dpkg `-ilibeigen3-dev_3.2.5-4_all.deb`
- sudo apt-mark hold libeigen3-dev
- wget `http://www.vtk.org/files/release/7.1/VTK-7.1.0.tar.gz`
- tar-`xfVTK-7.1.0.tar.gz`
- cd `VTK-7.1.0&&mkdirbuild&&cdbuild`
- cmake ..
- make
- sudo make install
- cd /Downloads
- wget `https://github.com/PointCloudLibrary/pcl/archive/pcl-1.8.0.tar.gz`
- tar -xf pcl-1.8.0.tar.gz
- cd `pcl-pcl-1.8.0&&mkdirbuild&&cdbuild`
- cmake ..
- make
- sudo make install

Installing MATLAB

1. Download it from the official website of Mathworks. 2. Run the setup as an administrator 3. Then select the products to install as per your requirements and then enter activation key to activate MATLAB.

A.2 Setting up Project on PCL

With C++ installed on your system, the below linked URL provides detailed guidelines to set up project on PCL on each system of your ubuntu 16.04 system Since C++ is already installed on the system, proceed directly to Step 2 in the tutorial.

Link:- `http://www.pointclouds.org/documentation/tutorials/using_pcl_pcl_config.php`