SOFTWARE ENGINEERING
PROJECT REPORT

SCANNO-3D

Authors:
Rahul Pandey - MAIA
Majeed Hussain - MSCV
Zafar Toshpulatov - MAIA

Supervisors:
Prof. Yohan Fougerolle
Cansen Jiang
David Strubel

UNIVERSITY OF BURGUNDY
2017 - 2018

Contents

1. *Overview*

The main goal of the project is to create a 3D scanning system capable of capturing the full body of a human. Complete 3D scanning systems available in the market for scanning full body of a human being consist of a turn-table that can support a person, as well as a rig for holding and moving the camera.

Figure 1.1: Application Workflow

The setup consists of a stationary camera. The person is captured at different orientations with respect to the camera and the data is stitched to create a 3D reconstruction. The reconstruction is desired in the form of a watertight mesh.

2. *Project Planning and Management*

Project planning and management is one of the key aspect of this project as there are 3 members in the group, each belonging to different nationalities and unique skill sets. This called for the use of proper software methodology in order to make the requirements achievable.

2.1 Development platform

2.1.1 Software requirement

1. Qt 5.7.0 (MSVC 2013, 32 bit)
2. MSVC 2015 build tools
3. PCL 1.8
4. Microsoft Kinect V2 SDK
5. Intel R200 SDK

2.1.2 Hardware requirement

1. Microsoft Kinect V2
2. Intel R200

2.2 Collaboration platform

Considering the size of the group different collaboration platforms were used for different purposes.

2.2.1 Github

Github is a web based Git repository system used for version control of documents and code. The URL for the github repository is `https://github.com/rahulpandeycs12/Software-Engineering_Project-3-D-Scanner.git`

It was mainly used in the project for the below purposes:

1. Version control of documents and code
2. Collaborative programming
3. Integration platform

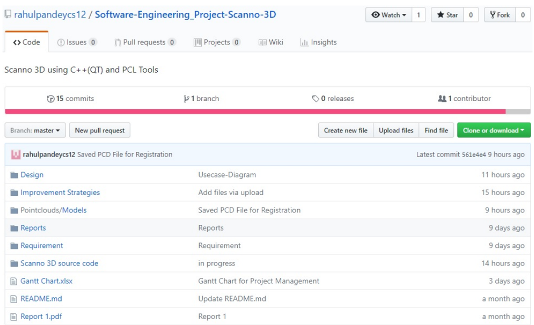


Figure 2.1: Github repository

3. *High level design*

3.1 Use case diagram

The diagram depicts the different features provided by the application at a high level and the types of users who can have access to the system.

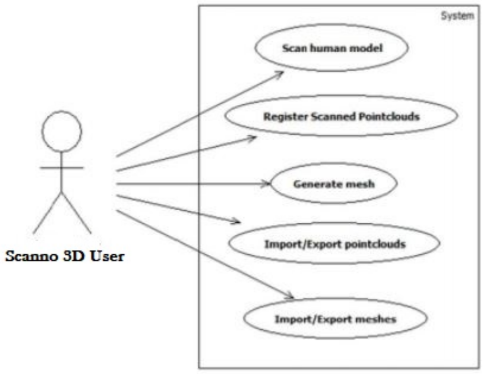


Figure 3.1: Use case diagram

3.2 Class diagram

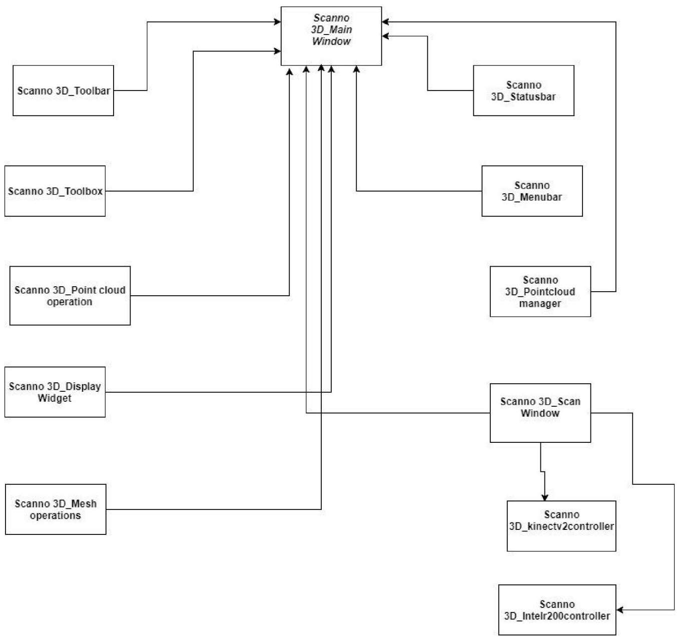


Figure 3.2: Class Diagram

4. *Pointcloud acquisition*

4.1 Introduction

Due to the fact that the project depends on a sensor, it is essential to have an acquisition module to obtain the desired sequence of data for all its posterior processing. In order to enhance the probabilities of being this setup useful for the final user, two sensors have been taken into account and development has been done for both:

1. Microsoft Kinect V2.
2. Intel R200.

5. *Registration*

5.1 The registration problem

The process of 3D data alignment is generally known as registration. Its corresponding algorithms addresses the problem of finding the optimal transformation that maps a pair of point clouds between them, considering that both datasets are exposed to camera noise during acquisition. Overall, there are two main families of registration methodologies: rigid and non-rigid ones. While the first family assumes that the mapping transformation can be modeled by using six degrees of freedom, the latter one considers the problem of softer bodies whose shape can change during data acquisition [Bellekens 2014].

5.2 First approach: General Case

Our first approach to the registration problem used pairwise registration based on feature correspondences. The original point cloud was initially downsampled to around 40% of the original data size in order speed up computation by means of data dimensionality reduction. Then, a transformation matrix was estimated with respect to the previous downsampled point cloud using features descriptors. Once pre-alignment was obtained, a final aligning step was conducted and the obtained transformation using downsampled versions was applied to the original raw point cloud to finish the pairwise registration. A flowchart of this registration approach is shown in Figure 5.1.

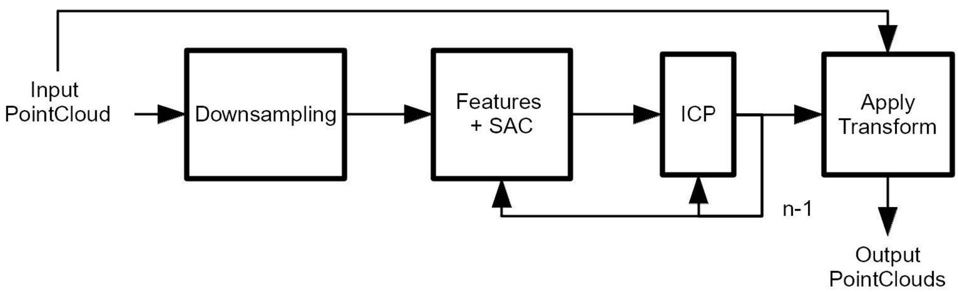


Figure 5.1: Chart flow for the first registration approach.

In the first step, a voxelized grid approach was conducted using the VoxelGrid class in order to downsample the input point cloud. Here, a custom size 3D voxel grid is created over the input cloud and all data inside each voxel is represented by means of its voxel centroid. As result, the input point cloud is described by a lower dimension 3D cloud that still preserves the high-dimension cloud morphology.

After downsampling, the correspondence matching between point clouds was eased by using Fast Pointer Feature Histograms (FPFH), which consists in informative and pose invariant local features describing the surrounding surface properties of a point [Rusu 2009].

For pre-aligning two point clouds based on its FPFH descriptors, Sample Consensus Initial Alignment (SAC) was conducted. The method consists in an exhaustive search in the correspondences

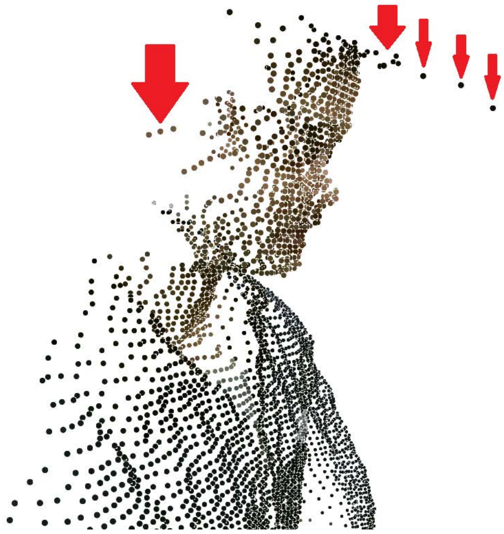


Figure 5.2: Noisy point cloud. Red arrows highlights outliers.

space (FPFH descriptors of the clouds) in order to find the best transformation that aligns data based on its features [Rusu 2009].

Finally, once a pre-alignment was achieved using FPFH and SAC methods, the Iterative Closest Point (ICP) algorithm was used to register the last point cloud with the previous one. The method uses singular value decomposition to estimate the affine transformation that better aligns data.

5.3 Denoising and Downsampling

To tackle this problem, statistical outlier removal was used by means of a k-nearest neighbours approach. Here, k-points surrounding a specific point of the cloud are analyzed. The statistical parameters of the sample distribution are estimated (mean and standard deviation), and those points falling out of m-times the standard deviations are removed. Hence, the algorithm requires setting the values k and m for the thresholding. In our implementation, values k=8 and m=2.5 were used.

5.3.1 Pairwise Initial Rough Alignment

After this pre-processing steps, the rough registration transform is estimated with a similar setup than that of the first approach. We first get a list of correspondent points between the new downsampled point cloud and the previous registered downsampled based this time solely on the normals as descriptors. Then a consensus correspondence rejection algorithm is applied to discard possible outliers. Afterwards, a linear system is solved that gives the best approximate transform to map each correspondence to the same frame of reference using SVD for efficient computation.

Finally, ICP is applied to the result of the previous step to get a finer alignment. The obtained transform is multiplied with the one from the previous step and applied to the original not-downsampled point cloud. At this point the rough initial alignment is done and we either wait for another pointcloud to register or for the user to start the post-processing.

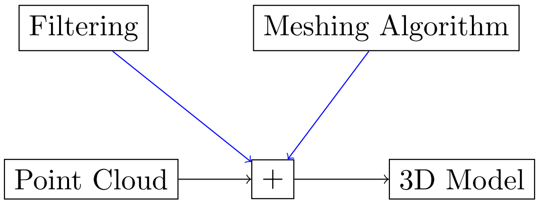
5.3.2 Post-Processing

The small error in pairwise registration have induced a significant error when the view returns to the initial position. We can see that the legs are significantly misaligned, causing the appearance of having three legs and two pairs of hands. This is a common problem in robotics and to solve this issue we used a variant of the well-known SLAM (simultaneous localization and mapping) technique through the ELCH (Explicit Loop Closing Heuristic) implementation in PCL. The result by slightly modifying the initial transforms until the last point cloud matches with the first one is a better global registration. However, this small modification induce at the same time small local misalignments as a consequence, which need further processing since ELCH goal is to provide good global registration and not so much good registration at the local level.

6. *3D Reconstruction*

6.1 The 3D Reconstruction Problem

Constructing a surface in the presence of these data anomalies is a difficult problem. In addition scanning technologies have driven a dramatic increase in the size of datasets available for reconstruction, with datasets now exceeding one billion point samples. As a result, space and time efficiency have become critical in the development of effective reconstruction algorithms surface reconstruction The reconstruction can be summarized like in scheme below:



3D RECONSTRUCTION

In our specific case the Point Cloud that we receive from the registration class was quite noisy, with a high density of point and we front even another problem "the extralayer of the surface" for better understand this problem we can see the following image:

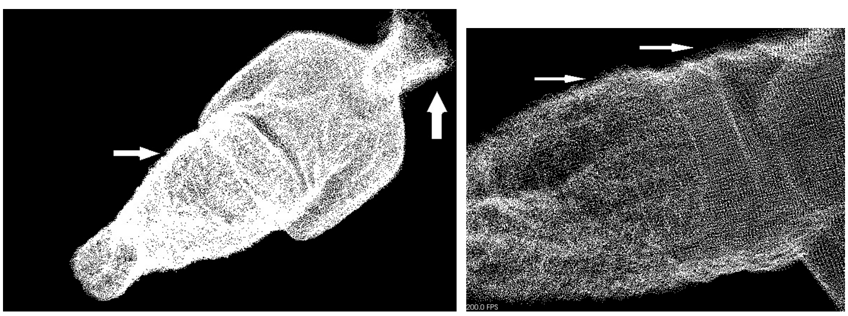
The different techniques used for making 3D reconstruction will be discussed and tested from different aspects.This is done in order to find the most optimal reconstruction technique.

6.2 Preparing the cloud

Before going into the implementation of the algorithms : greedy triangulation, marching cube and Poisson,it is necessary to look at the input data that is provided to these algorithms and how it is processed, so before give the point cloud tho this algorithm we have to do some operation on the point cloud:

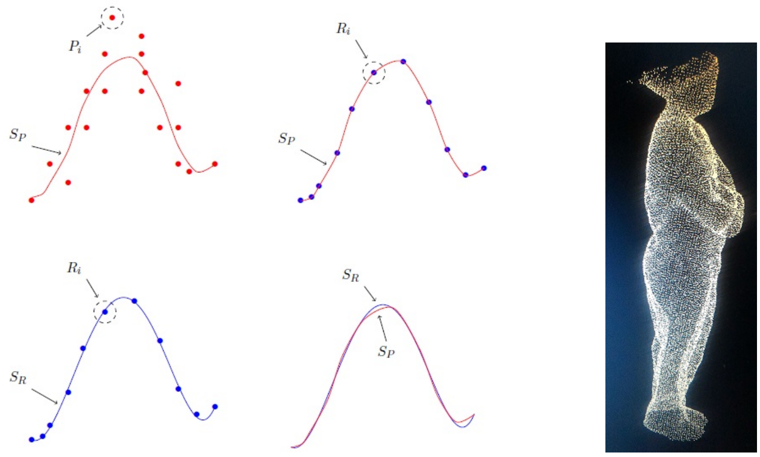
6.2.1 Filtering

Because the received point cloud it was oversampled first of all we decide to elaborate on it some filtering operation:



(a) high density point cloud (b) extralayer of the surface

Figure 6.1: Common issues for 3D reconstruction



(a) MLS (b) Point Cloud after MlS

Figure 6.2: MLS Process

- PassThrough this filter iterates through the entire input Point Cloud,and automatically filtering the non-finite points and the points outside to one presetted interval specified
- The VoxelGrid class creates a 3D voxel grid, as a set of tiny 3D boxes in space, over the input point cloud data. Then, in each voxel all the points present will be approximated with their centroid. This approach is a bit slower than approximating them with the center of the voxel, but it represents the underlying surface more accurately.

6.2.2 MLS smoothing

The first step taken after preparing the cloud is to smooth it using Moving Least Squares(MLS). As explained by [Alexa, Behr, Cohen-Or, Fleishman, Levin, and Silva, 2003] the idea behind MLS is to have a data set of points $P = \{pi\}$ that constructs a smooth surface SP , however, instead of using the original data set of points P, a reduced set $R = \{ri\}$ is created and a new surface SR is defined.

The MLS smoothing is done before any reconstruction algorithm is applied. Another smoothing method is also used, however only applied after the reconstruction is complete.

6.3 Poisson

There are a multitude of surface reconstruction techniques, however each one of them poses a number of difficulties when it is applied to data points. But because watertight mesh was one of the prefixed setted point,at the end, we decided to use Poisson Algorithm Reconstruction. The Poisson reconstruction uses the Poisson equation, which is also known to be used in systems that perform tone mapping, uid mechanics and mesh editing. The Poisson reconstruction approached by Michael Kazhdan, Matthew Bolitho and Hugues Hope has set as goal to reconstruct a watertight model. It need in input the normal direction, respect to the surface, for each points of the cloud.

By its nature, this algorithm can only be used with watertight or closed object. This can be problematic when we get the points cloud through Kinect, in fact the acquisition that we make not always respect this Property, as we can see in our model the head is not a close object. In our specific case at the head and feet can be noticed that the Poisson reconstruction connects the regions, which

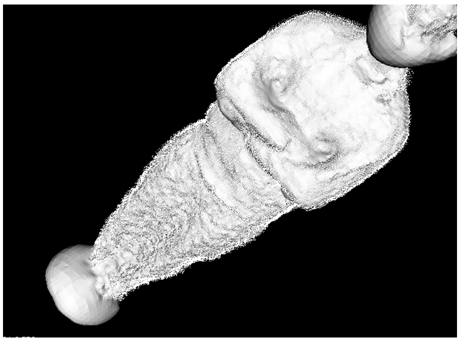
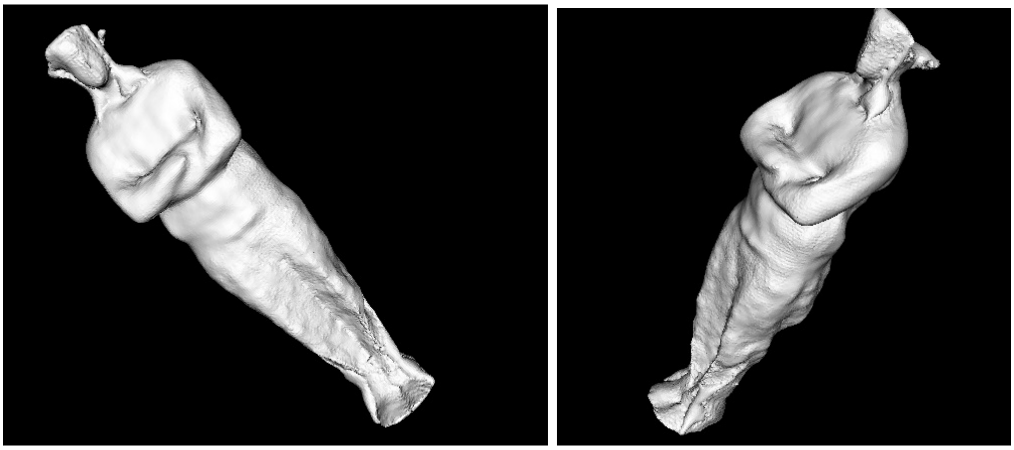


Figure 6.3: Poisson Reconstruction



(a) Poisson after Laplacian Filter (b) Poisson after Laplacian Filter

Figure 6.4: MLS Process

poses some limitations in the use of Poisson reconstruction, even more Poisson reconstruction has a high noisy sensitivity, we can see this in the images below:

6.4 Postprocessing algorithm Laplacian

The obtained result was still a lot noisy, for having a better result in terms of surface problem we decide to apply a post smoothing method.

The smoothing method used is MeshSmoothingLaplacianVTK. This is a laplacian smoothing method from the VTK library. As described in the documentation for the original VTK smoothing method: *"The effect is to "relax" the mesh, making the cells better shaped and the vertices more evenly distributed". [Visualization Toolkit*, 2015].

7. *Graphical User Interface*

This chapter will introduce the Graphic User Interface (GUI) developed, its functionality and an analysis of the structure and the different mechanisms. The GUI has been divided into 2 independent windows, namely $EditWindow$ and $ScanWindow$.

The functioning of the window is based on its 4 different components which will be detailed below.

7.1 Menu bar

The menu bar (figure 7.2), located in the upper left corner, give the possibility to the user to import and export points clouds and meshes.

The point clouds, under the format .ply or .pcd, could be imported from the local computer or directly from the scanning interface (Section 7.4). The meshes, under the format .stl and .vtk could be also import from the local computer or generating using the module *PointCloudOperations*. By importing from the local computer, the possibility to import files at the same time has been added.

To export a point cloud or a mesh, its required to specify the format of the file by selecting it from the sub menu *ExportAs*.

7.2 Point Cloud Operations

Different operations can be processed to the point clouds such as the registration between several point clouds and the generation of a mesh.

The Point Cloud Operations module (figure 7.4) allows the user to choose the algorithms of registration and generation of mesh. By default, these algorithms are respectively Singular Value Decomposition and Iterative Closest Point for the registration and Poisson Surface Reconstruction for the generation of mesh. When one of these operations is applied, the generated file is directly added to the QListWidget of the *PointCloudExplorer*.

7.3 Point Cloud Visualizer

The Point Cloud Visualizer (figure 7.5 module is based on a QVTKWidget which give us the possibility to display a point cloud or a mesh.

Each time a QListWidgetItem of the *PointCloudExplorer* is checked or unchecked, the point cloud or mesh associated is added or removed to the QVTKWidget. Multiples files can then be displayed at the same time. In order to have a better visualization, its possible to zoom in / zoom out and rotate around the axis [x, y, z].

7.4 Scan Window

From the Scan Window (figure 7.6, the user will have the possibility to interact directly with the sensor and the platform to get a series of point clouds of the scanned person, which will be display on the *Point Cloud Visualizer* module, similar to the one of the Edit Window (section 7.3).

Figure 7.1: Preview of the Edit Window

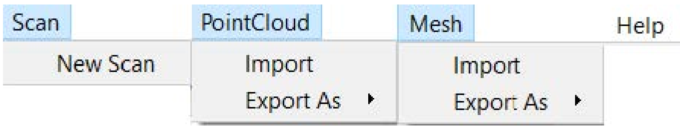


Figure 7.2: Preview of the Menu Bar

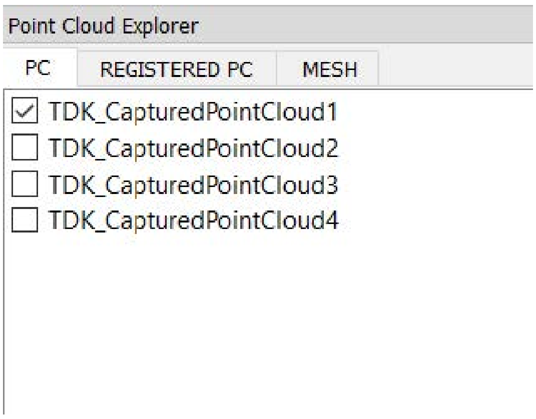


Figure 7.3: Preview of the Point Cloud Explorer

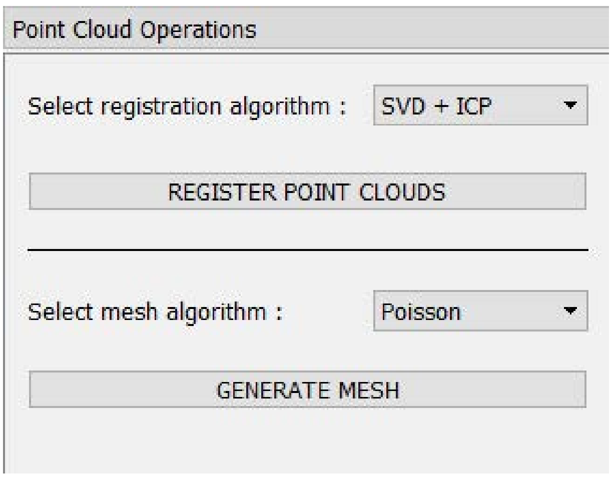


Figure 7.4: Preview of the Point Cloud Operations

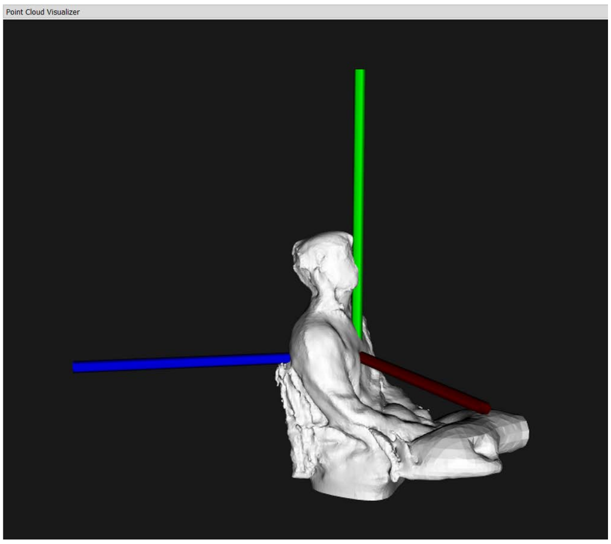


Figure 7.5: Preview of the Point Cloud Visualizer while displaying a mesh

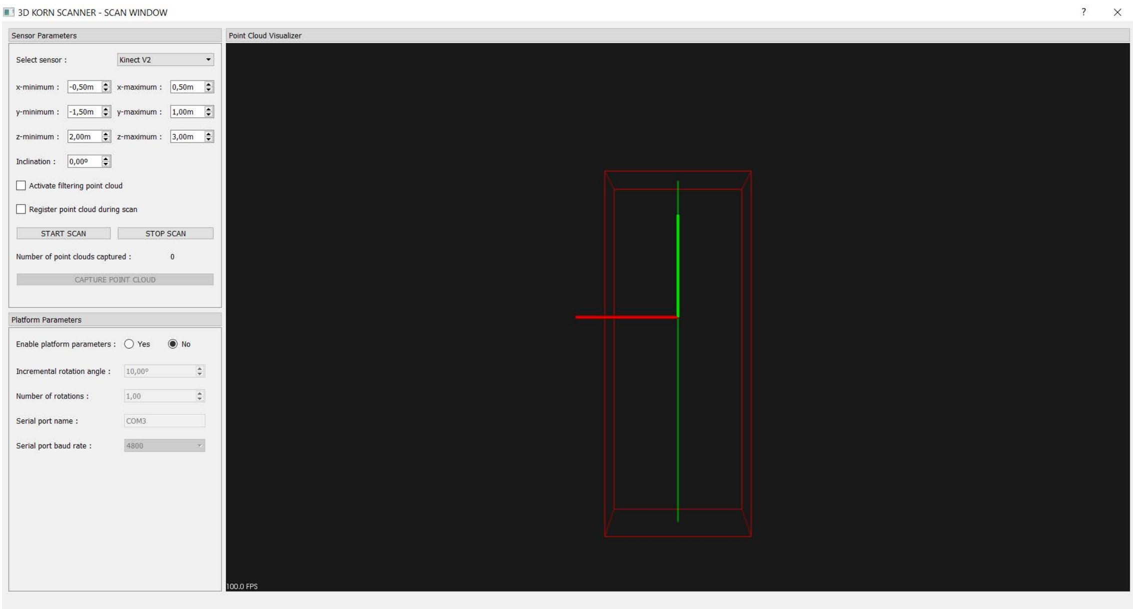


Figure 7.6: Preview of the Scan Window

8. *Improvements in Registration phase of 3D Scanning*

The previous work that was done in registration phase of 3D Scanning was based on traditional approach and existing registration method. Here in this report we have tried to propose a new approach which if used with registration phase gives more efficiency and effectiveness to the results.

3D point cloud registration is a fundamental and critical issue in 3D reconstruction and object recognition. Most of the existing methods are based on local shape descriptor. Here, we propose a Regional Curvature Map (RCM) method. Based on the RCM, an efficient and accurate 3D point cloud registration method is presented. We firstly find 3D point correspondences by a RCM searching and matching strategy based on the sub-regions of the RCM. Then, a coarse registration can be achieved with geometrically consistent point correspondences, followed by a fine registration based on a modified iterative closest point (ICP) algorithm.

The experimental results demonstrate that the RCM is distinctive and robust against normal errors and varying point cloud density. The corresponding registration method can achieve a higher registration precision and efficiency compared with two existing registration methods.

8.1 Traditional point cloud Alignment

Suppose an object requires a 360-degree 3D digitization.

An initial point cloud is acquired from a first view point. A second point cloud is acquired from a different viewing angle. If the relative camera position is unknown the two-point clouds cannot be easily aligned with current technology.

As any registration problem, range registration consists of the steps of matching and estimation of the rigid transformation. Depending on the displacement and orientation between point clouds, we differentiate between crude and fine. alignment. The challenge in crude registration lies in performing it automatically and consistently even when there is very small overlap. The golden standard for fine registration is the Iterative Closest Point algorithm and its variants. ICP techniques either assume a rough alignment of the two-point sets or run the algorithm multiple times by sampling the space of initial conditions. Robust variations like RANSAC reject outliers and improve the estimation of the rigid transformation.

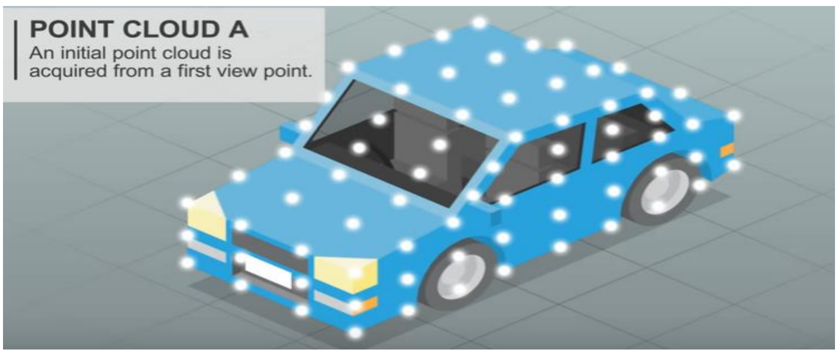


Figure 8.1: Initial Point Cloud A

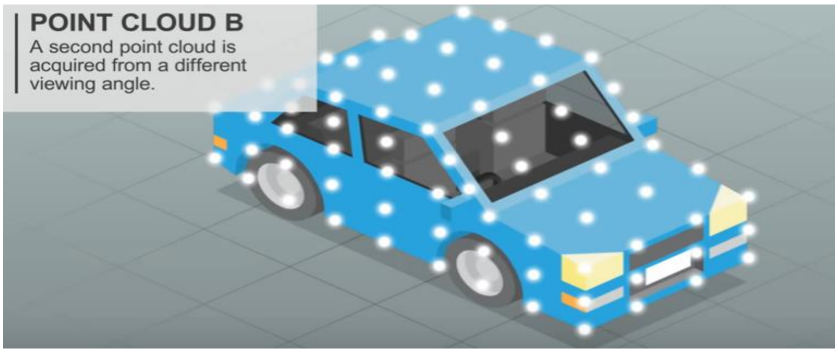


Figure 8.2: Initial Point Cloud B

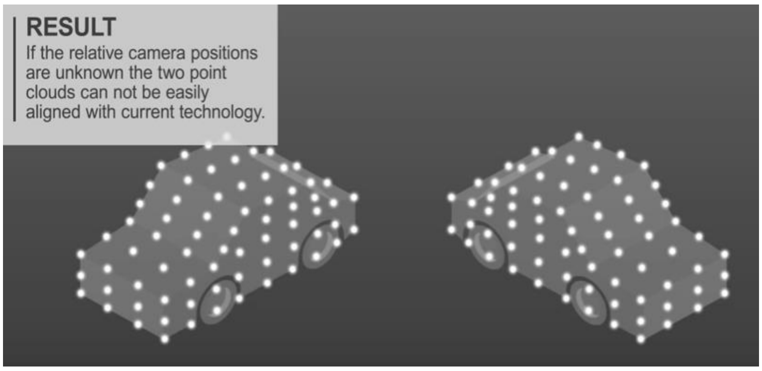


Figure 8.3: Two frames of object

Traditionally a human user must identify a series of points in each point cloud that represent the same part of the real-world object.

This process has the potential to be time consuming and incorrect. It also becomes impractical processing multiple point clouds such as in video. Once the overlapped areas have been identified it is a simple computation to align the point clouds to produce a complete model of the object.

At this stage the optimization needs to be done and so the alignment can be refined with the common ICP algorithm.

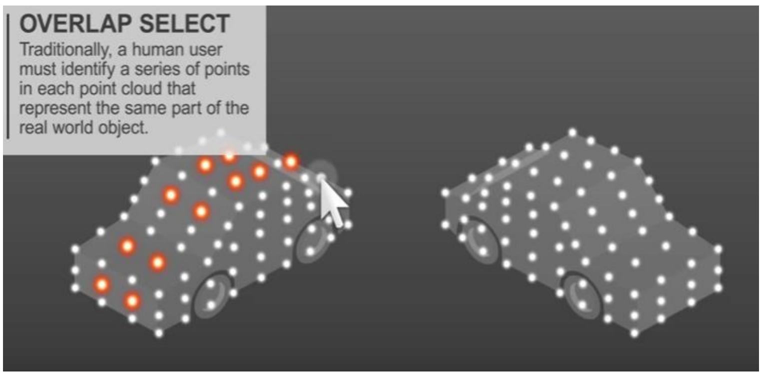


Figure 8.4: Identification of Series of point on Cloud A.

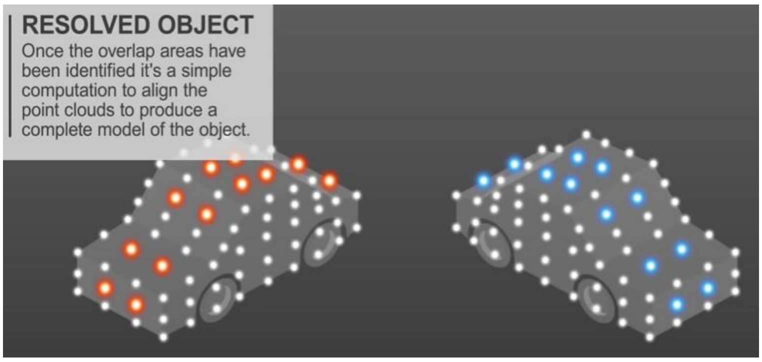


Figure 8.5: Identification of Series of point on Cloud B.

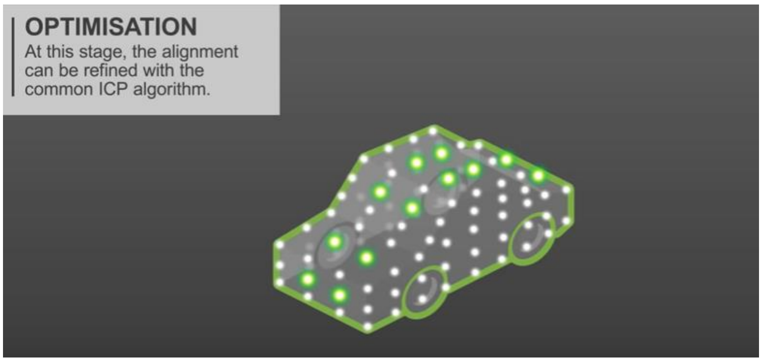


Figure 8.6: Optimization

8.2 Proposed Modern Technique for Point Cloud Alignment

In a different technique the 3D point cloud registration can be done using identifying normal. First the surface normal are calculated for each point, in all point clouds, the normal are collected on a common origin forming clusters on a sphere.

Clusters of normal tend to represent a surface, and the pattern of clusters represent part of the object. Comparing the two patterns will reveal the angles that are coherent between point cloud A and point cloud B.

This comparison allows the angle of rotational difference between the two-point clouds to be calculated.

The calculated angle of difference is then applied to one of the point clouds, matching its rotational orientation with the other. The final step is simply to translate one-point cloud over to the other until

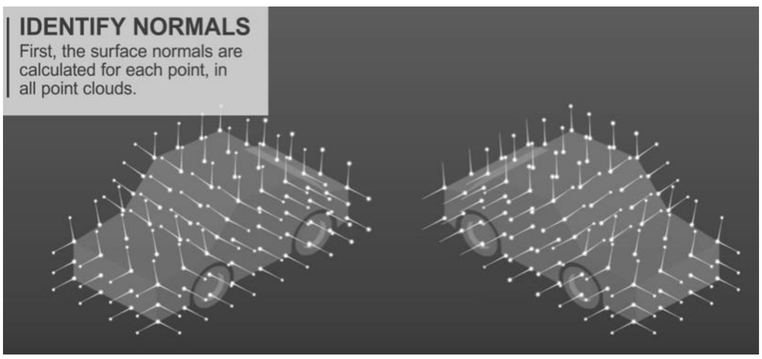


Figure 8.7: Surface Normal for all Points

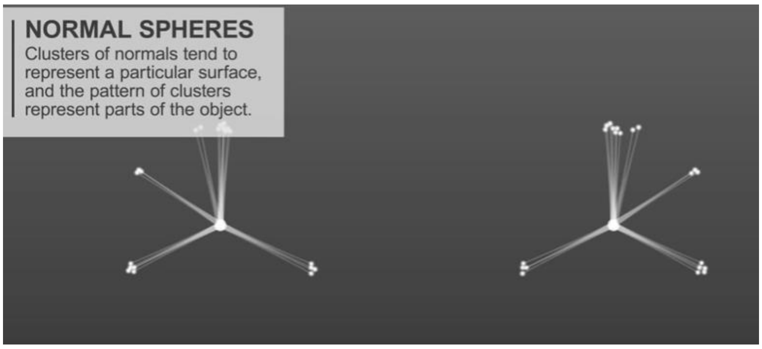


Figure 8.8: Clustering at Origin

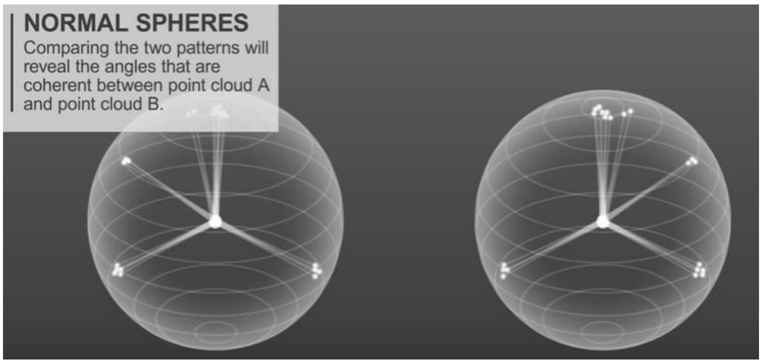


Figure 8.9: Comparison of Normal Spheres

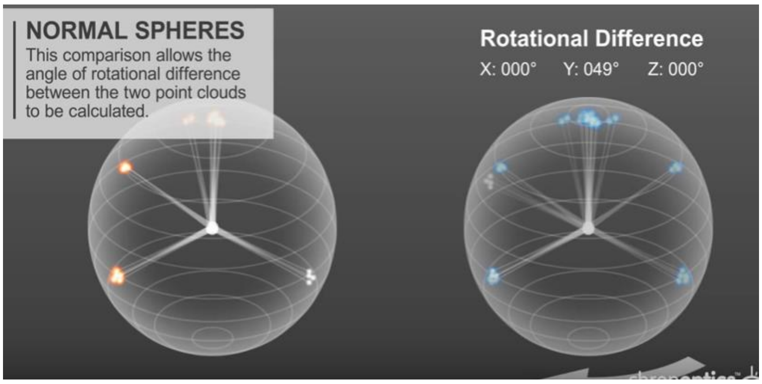


Figure 8.10: Rotational Difference Calculation

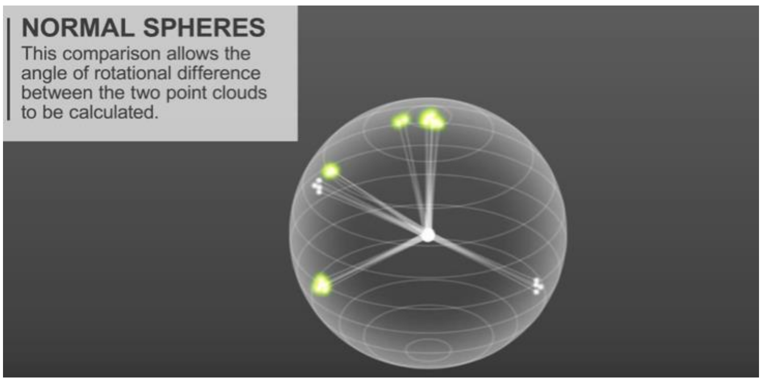


Figure 8.11: Overlapping of two Normal Spheres

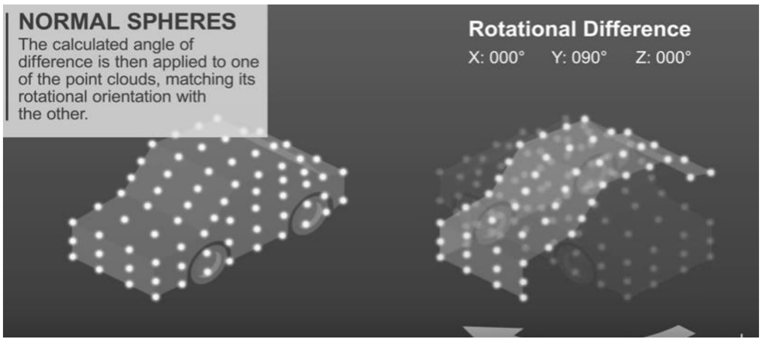


Figure 8.12: Difference angle applied to point clouds

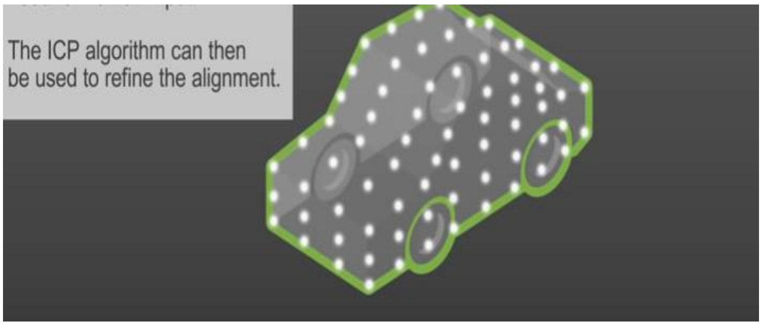


Figure 8.13: ICP algorithm for refinement

it overlaps.

The result is a course alignment of point clouds with large unknown separation angles. The ICP algorithm can be then used to refine the alignment

8.3 Changing outlier

There are several different methods in the filter module to remove outliers from a Point Cloud. In the previous case statistical outlier removal method was used which works fine for outlier removal. For the improvement purpose we look at how to use a Conditional Removal filter which removes all indices in the given input cloud that do not satisfy one or more given conditions. Then we tried how to use a RadiusOutlierRemoval filter which removes all indices in its input cloud that dont have at least some number of neighbors within a certain range.

The final output shows no improvement in terms of efficiency and precision for outlier removal and it seems that Conditional Removal filter and RadiusOutlierRemoval filter are not the best methods to use for outlier removal.

8.4 Performance Analysis of Rigid 3D Pointcloud Registration Algorithms

The registration algorithms can be classified coarsely into rigid and non-rigid approaches. Rigid approaches assume a rigid environment such that the transformation can be modeled using only 6 Degrees of Freedom (DOF). Non-rigid methods on the other hand, can cope with articulated objects or soft bodies that change shape over time.

Most of these employs either a simple Singular Value Decomposition (SVD) or Principal Component Analysis (PCA) based registration, or use a more advance iterative scheme based on the Iterative Closest Point (ICP) algorithm. Recently, many variants on the original ICP approach have been proposed, the most important of which are non-linear ICP, generalized ICP, and non-rigid ICP. The choice for one of these algorithms generally depends on several important characteristics such as accuracy, computational complexity, and convergence rate, each of which depends on the application of interest.Therefore, in this report we discuss the mathematical foundations that are common to the most widely used 3D registration algorithms, and we compare their strengths and weaknesses in different situations.

8.4.1 Registration algorithms

Here, we discuss five widely used rigid registration algorithms. Each of these methods tries to estimate the optimal rigid transformation that maps a source point cloud on a target point cloud. Both PCA alignment and singular value decomposition are pairwise registration methods based on the covariance matrices and the cross-correlation matrix of the pointclouds, while the ICP algorithm and its variants are based on iteratively minimizing a cost function that is based on an estimate of point correspondences between the pointclouds

A Principal Component Analysis:

PCA is often used in classification and compression techniques to project data on a new orthonormal basis in the direction of the largest variance. The direction of the largest variance corresponds to the largest eigenvector of the covariance matrix of the data, whereas the magnitude of this variance is defined by the corresponding eigenvalue. Therefore, if the covariance matrix of two pointclouds differs from the identity matrix, a rough registration can be obtained by simply aligning the eigenvectors of their covariance matrices.

B Singular Value Decomposition:

PCA based registration simply aligns the directions of the largest variance of each pointcloud and therefore does not minimize the Euclidean distance between corresponding points of the datasets. Consequently, this approach is very sensitive to outliers and only works well if each pointcloud is approximately normally distributed. However, if point correspondences between the two pointclouds are available, a more robust approach would be to directly minimize the sum of the Euclidean distances between these points. This corresponds to a linear least-square problem that can be solved robustly using the SVD method.

C Iterative Closest Point:

Point correspondences between these pointclouds are defined based on a nearest neighbor approach or a more elaborate scheme using geometrical features or color information.

SVD, is used to obtain an initial estimate of the affine transformation matrix that aligns both pointclouds. After registration, this whole process is repeated by removing outliers and redefining the point correspondences. Two widely used ICP variants are the ICP point-to-point and the ICP point-to-surface algorithms. These approaches only differ in their definition of point correspondences.

- **ICP point-to-point:** The ICP point-to-point algorithm was originally described in [1] and simply obtains point correspondences by searching for the nearest neighbor target point qi of a point pj in the source pointcloud.
- **ICP point-to-surface:** The ICP point-to-surface algorithm assumes that the point clouds are locally linear, such that the local neighborhood of a point is co-planar. This local surface can then be defined by its normal vector n, which is obtained as the smallest eigenvector of the covariance matrix of the points that surround correspondence candidate qi.

D Generalized ICP: A major disadvantage of the traditional point-to-point ICP algorithm, is that it assumes that the source pointcloud is taken from a known geometric surface instead of being obtained through noisy measurements. However, the point-to-surface ICP algorithm relaxes this constraint by allowing point offsets along the surface, to cope with discretization differences. However, this approach still assumes that the source pointcloud represents a discretized sample set of a known geometric surface model since offsets along the surface are only allowed in the target pointcloud. To solve this, Generalized ICP algorithm is proposed which performs plane-to-plane matching. They introduced a probabilistic interpretation of the minimization process such that structural information from both the source pointcloud and the target pointcloud can be incorporated easily in the optimization algorithm. Moreover, they showed that the traditional point-to-point and point-to-surface ICP algorithms are merely special cases of the Generalized ICP framework.

8.4.2 Results and discussion

We illustrate the performance difference between a naive PCA based approach, a correspondence based SVD approach, and the ICP point-to-point registration approach. Figure shows the matching error plotted against the number of iterations for the ICP point-to-point algorithm without pre-alignment, and for the ICP point-to-point algorithm (light-gray) where the data has been pre-aligned using the SVD approach. In the latter case, a simple nearest neighbor matching was used to define point correspondences, after which the SVD algorithm was used to solve the least squares problem. This result clearly shows the importance of a rough initial alignment before applying the ICP algorithm.

Furthermore, figure shows the results of a single SVD based least-squares iteration, and the results obtained using the PCA based registration approach. The PCA based approach yields the largest matching error, since it does not incorporate correspondence information, such that this method is highly sensitive to outliers. On the other hand, a simple PCA or SVD based approach is extremely computational efficient, whereas the iterative ICP scheme is often too computationally expensive for real-time applications. However, Figure 7 shows that convergence can be reached quickly if a rough initial alignment is available. Finally, it is important to note that result of the variants of ICP such as point-to-plane and plane-to-plane greatly depend on the input data. If the source pointcloud does not

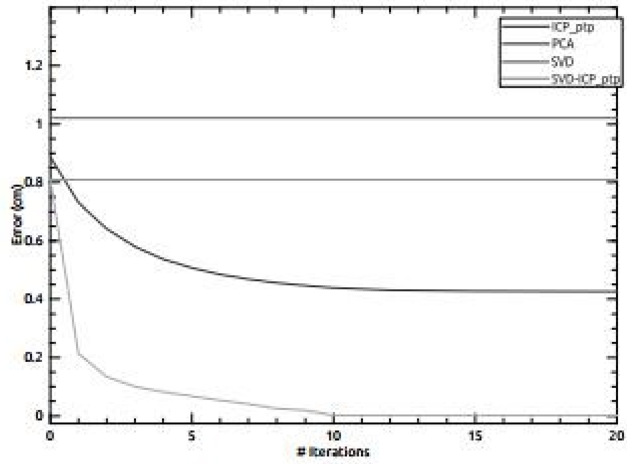


Figure 8.14: Comparison between PCA, SVD and general point to point ICP

contain much noise, while the target pointcloud is mostly smooth and piece-wise planar, the point-to-plane algorithm outperforms. the traditional point-to-point method. On the other hand, if the geometric structures in the scene are mostly quadratic or polynomial, the traditional ICP point-to-point algorithm yields better results. Similarly, if a lot of noise is observed in the source pointcloud, ICP plane-to-plane outperforms ICP point to-plane.

8.5 Results and conclusions

8.5.1 Results

The results obtained are really good at the global registration but with a lot of room for improvement in local registration, having a good deal noise and artifacts. The artifacts and noise in the attached figures may probably be due illumination issues and the noise in the sensor itself, they will be removed with smoothing filters and other post processing for reconstruction.

8.5.2 Conclusions and Future work

We consider the results of our registration algorithm to be good enough considering the technical difficulty, initial skills and knowledge, available time and mentoring and organisation overhead in the project. We have provided an algorithm that can successfully register 3D objects with as few as 10 frames while providing a rough registration preview in real-time. It is also robust to rough parameter estimation and errors in the pre alignment of some frames. The developed class is modular and flexible for ease of future development, while offering a simple use interface and configurable parameters.

Future work should focus mainly on reducing the noise and misalignment derived from sensor noise and non-rigid object registration. The severity of the sensor noise in the registration process has only been noticed in the final stages of development and it hasnt been fully addressed. Consequently, the outlier removal step and point cloud denoising should be further studied and developed. However, even with really good outlier removal and noise reduction, if the scanned person makes small movements between frames (such as breathing), it will still impact the quality of registration for mesh

Figure 8.15: Meshing result 1 and Meshing result 2

reconstruction. It is suggested to study the integration of non-rigid fine registration algorithms that can account for this misalignments and can adjust them to create a smooth yet detailed final result.

References

1. *A survey of rigid 3D pointcloud registration algorithms;* Bellekens, B., Spruyt, V., Berkvens, R., Weyn, M.

2. *Fast Point Feature Histograms (FPFH) for 3D Registration;* R. B. Rusu, N. Blodow, and M. Beetz

3. *KinectFusion: Real-Time Dense Surface Mapping and Tracking;*

4. *DynamicFusion: Reconstruction and Tracking of Non-rigid Scenes in Real- Time;*

5. `http://doc.qt.io/`

6. `http://pointclouds.org/documentation/`

7. `http://www.cplusplus.com/`

8. `https://github.com/UnaNancyOwen/KinectGrabber`

9. `https://software.intel.com/sites/landingpage/realsense/camera-sdk/v1.1/documentation/html/index.html?doc_devguide_introduction.html`

10. `https://codeyarns.com/2015/11/26/how-to-use-kinect-v2-on-windows/`

11. `https://www.udacity.com/course/how-to-use-git-and-github--ud775`

12. `http://www.vtk.org/Wiki/VTK/Tutorials/QtSetup`

13. `https://en.wikipedia.org/w/index.php?title=Point_set_registration&oldid=732521446`

14. A. Adan, C. Cerrada, and V. Feliu. Global shape invariants: a solution for 3D free-form object discrimination/identification problem. Pattern Recognition, 34:1331 1348, 2001.

15. G. Arfken and H. Weber. Mathematical Methods for Physicists. Academic Press, 1966.

16. G. Arfken and H. Weber. Mathematical Methods for Physicists. Academic Press, 1966.

17. J. A. Beraldin, L. Cournoyer, M. Rioux, F. Blais, S. F. ElHakim, and G. Godin. Object model creation from multiple range images: acquisition, calibration, model building and verification. In International Conference on Recent Advances in 3-D Digital Imaging and Modeling, 1997.[1] A. Adan, C. Cerrada, and V. Feliu. Global shape invariants: a solution for 3D free-form object discrimination/identification problem. Pattern Recognition, 34:1331 1348, 2001.