UNIVERSITÉ DE BOURGOGNE

Software engineering

January 8, 2018

3D Scanner Project

Team Members:

BATERIWALA, Malav

OCHOA, Eduardo

VAISHNAV, Mohit

VALENCIA, Liliana

Project Supervisors:

Dr FOUGEROLLE Yohan

Dr CANSEN Jiang

STRUBEL David

Contents

List of Tables

List of Figures

1 Introduction

The 3D Scanner technology allows the collection of data from the real world to construct digital 3D models. These models try to reconstruct the color and shape of the objects, environments or human body as close as possible of the reality.

Human body scanning has gained special attention due to its application, so that the technology is more accessible and affordable and it is not longer restricted to the industry or research.

As the demand of this technology increase, the possibility of having a software that allows user to perform 3D scanning in short period of time and with very good result also increase. To make it possible, a good quality of tools are available in hardware as Microsoft Kinect Sensor and software as Qt -*cross-platform applicationframework*- and libraries as PCL -*Point Cloud Library*- and OpenCV -*Open Source Computer Vision Library*-.

Due to the development of *Kinect* from Microsoft for its Xbox games there is a rise in popularity of RGB-D cameras among researchers. Two versions of Kinect had been released with V2 using modulated light technology for depth measurement thereby producing better resolution and quality. Other advantages of using Kinect includes the affordability which enables its widespread use. It was initially designed to track the human movements which is a real time process where depth frames are captures at the rate of 30 frames per second. This high rate becomes one of the issue leading to lower accuracy and noisy image.

The aim of this project is to improve the acquisition and processing software built by the student of the last year, focusing in the performance's improvement and the integration of OpenCV.

Organization of the report is as follows. Objective [2] presents the aim and scope of the project. Project Management [3] elaborates about the distribution of the tasks amongst the group members to have the maximum output from them and utilize their capabilities in the best possible means given the short period to cover this

bulky project. Next section [4] enlists the various requirements to carry out this project and give a brief description. Overview of the project [5] describes the project in the most simpler terms to have a better understandable nature. It also provides the flow of the project which is needed to be carried out in addition to describing other available techniques in literature. Improvements [6] section explores more into the contribution made to carry out this project which gives the instructor a glimpse of the effort made by the group members. Finally Conclusion[7] is given in the last section of the report followed by the References.

2 Objective

The main objective of this project is to improve the code given from the students of last year focusing in the details that can improve the process performance and final result.

The implementation of this objective began by conducting a background research about the existing methods for acquiring depth data of the physical objects and about integrating multiple scans to form a solid 360-degree presentation. As a result, the implementation of this work is able to obtain depth scans of the objects positioned on planar surfaces, filter the background data out from it and merge multiple scans into a single point cloud presentation of the original object.

3 Project Management

For this project, a group of four people was made. During the entire development of the project, the leadership of the tasks are divided in order to have a better understanding in the team.

The main objective of the group is to be able to have an integration between OpenCV and Qt, specially in data acquisition and feature matching.

Considering the skills of each member of the team, they were put to accomplish the following below mentioned sections along with the participation in rest of the parts:

Member	Task to Lead
Malav	Coding
Eduardo	GUI
Mohit	Research
Liliana	Management

Table 1: Task leadership

In order to have a better control and development of the work, some of the tools were used which are as follows:

- *Slack*: Communication, planning, assignment of task and meeting set up. This app is also used as main communication with the supervisors. https://hakunamatataub.slack.com
- *Github*: Source code management, publishing week's reports. https://github.com/bigmb/HakunaMatata.git
- *GoogleCalendar*: A link between Slack and Google calendar was made to have the tracking of the tasks assigned.

The project schedule is established using a time line diagram where the main tasks are defined as well as the time duration of each one. A complete description of the tasks have been made in the weekly reports. Figure 1.

4 Software and Hardware Requirements

4.1 Software Requirements

To be able to run and build the project, it is necessary to install some programs, libraries and tools. They are listed as follows:

- QT Creator 4.0.3 with MSVC 2015
- VTK 7.0.0
- PCL all in one 1.8 installer
- Cmake
- VisualStudioCPPTools
- OpenCV libraries.
- Microsoft Kinect API

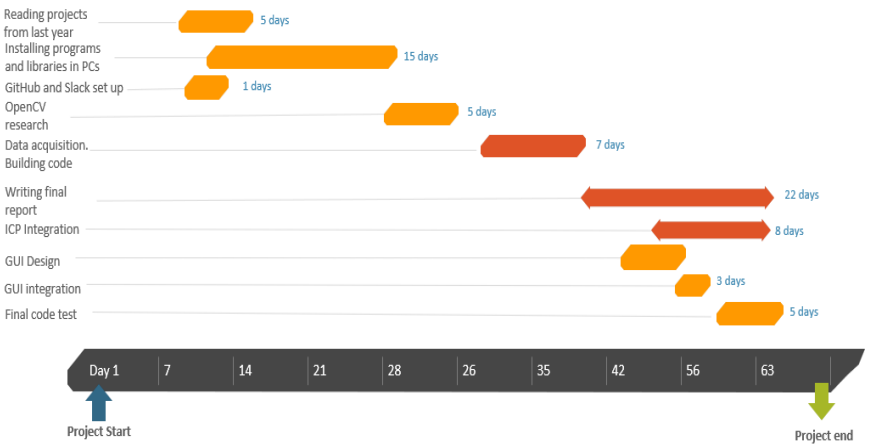


Figure 1: Time line Diagram

4.2 Hardware Requirements

The Microsoft Kinect V2 have been used as the only hardware for this project.

5 Overview of project

The main objectives of the project are:

- Acquire set of depth map with the object of interest.
- Enhance the depth maps and separate for the object of interest.
- Create an accurate $3D$ point cloud for object of interest.

To perform these tasks we have to take help from the previous year's project and create an efficient model this year making some improvement along with learning things given a short span of time. Some of the questions for research are:

- What improvements to be suggested for this year's project.
- Future scope of continuing our implemented work.
- How good our technique is, when compared to the previous ones.

Advantages of using Kinect as a sensors for research are:

- It simultaneously generate depth and color at standard video rate
- Using infrared, the depth sensor does not interfere with visual spectrum to some extent.
- Low cost, Reliability and speed of measurement.
- Dense reconstruction

While on the other hand some of its drawbacks are:

- Low X/Y resolution and Depth accuracy.
- Depth maps contains of numerous holes where there is no measurement.
- Limitation of view
- Completely fail on transparent surface object common in household
- calibration of Kinect sensor.

3D reconstruction using single Kinect involves the following steps:

- Surface measurement
- Camera poses estimation
- Fusion of aligned surface measurement.
- 3D model representation.

For reconstructing any kind of object to create a 3D model there are numerous steps involved. For example, in RGB-D framework commonly known two steps are: alignment of consecutive data frames, loop closure detection and global consistent alignment optimization. It aligns current frame to previous frame using frame to frame strategy. In alignment step it uses RGBD-ICP.

The process chain begins with obtaining input from Kinect sensor which is being held with the user directed towards the object of interest. This input is then down-sampled and separated from rest of the data to be presented for the user.

5.1 Acquisition

Microsoft Kinect is required for obtaining the input of the data. Some of the toolkit required are OpenNI which does the task of merging RGB data to depth and build a point cloud eventually to be visualized using cloud viewer functionality of PCL.

5.2 Filtering

Data filtering is required for the following reasons:

- *Segmentation*: Captured object needed to be separated from other unused data.
- $Performance$: Captured clouds have to be sub-sampled to obtain real time performance.

Filtering is also the limiting factor in capturing of object during real time. Because as and when the data is acquired from the Kinect all the process of filtering needed to be applied to the cloud. While it is done, rest input cloud are rejected by the system. The Filtering process can be done either in image space or in the Point clouds data.

5.2.1 Depth Filtering

Depth data obtained by the Kinect is in the range of 0.5m to 15m which increases as well the error in estimation. A simple pass through filter is used for filtering the depth data. It discards all the depth values greater than a particular threshold value for all depth field points.

5.2.2 Downsampling

After depth filtering, downsampling is performed to decrease the further computations. For this uniform sampling is used where 3D Voxel grid is created over input data after which in each voxel all the points are approximated by their centroid to get a single point which represents that voxel area.

5.2.3 Planar Removal

In our implementation the object is positioned in a planar surface, hence the points representing the object are to be separated from

the surface. It eases the process of object capturing. RANSAC (Random Sample Consensus) is used for detecting the largest planar component and filter it from rest of the data and performs this by finding inliers in the sampled data. Here it assumes that the floor or tabletop where object is placed is the largest planar component and leaves the other planar surface intact. But it could be problematic for scenarios where the object is standing in front of wall.

5.2.4 Clustering

Even after the planar removal there is a presence of unwanted objects. For filtering those background things we assume it to be consisted of object clusters where each of it is collection of a neighbouring points separated by other such similar cluster.

To implement this Euclidean Cluster Extraction algorithm of the PCL is applied. After they are identified, largest of the clustered is said to be object while the others are discarded.

5.2.5 Thresholding

If we build the system using the $OpenCV$, way which is to be followed is to first get the $Thresholding$ done. For calculation of $Threshold$ we calibrated the $Kinect$ sensor after which we find out the depth range of the image captured in the map. This depth data is then converted to the normal MKS measuring system using the calibrated $Kinect$ sensor coordinates to further discard the extra objects.

5.2.6 Feature matching

Features are one of the most important things in a registration process, so give the most important one to the algorithm will be very important for combining two different Point Clouds. The performance of the software and the convergence of the ICP method, that is done to align different frames of a rigid body and make a 3D model, will improved depending of how good features we have. So, how $OpenCV$ is an advantage to the 3D registration process?

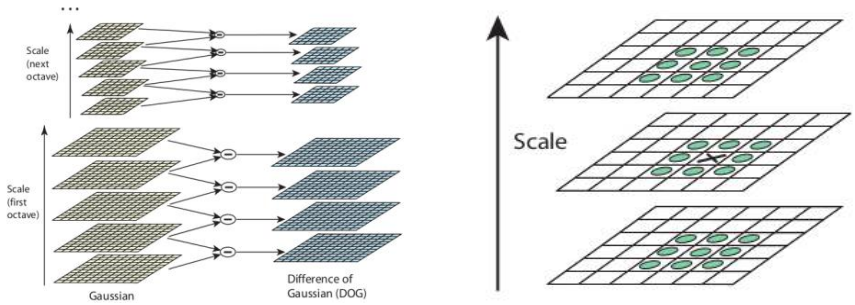
If two different frames of an image (with some degree of rotation or translation) are wanted to be stitched together then it is necessary to have specific patterns, that are unique in the scene, to

track and compare them with easiness. These patterns are the ones called $features$, and as humans they can be recognized easily but computers need some kind of parameters and information that tell them how to look for this kind of patterns.

In feature matching there are two kind of concepts that are important, and that are useful when it comes to the programming part in *OpenCV*. The first one is called *Feature detection*. This concerns to the specific patterns that the computer needs to stitch two images or point clouds together, and this can be given with ease in *OpenCV* via image analysis, in where the features are those regions in which they have maximum variation when moved [1]. The second important concept relates on how to find the same features in every image even when they have been moved. This is called *Feature description* and basically consist on giving some information relating to the feature found. This information is given by describing the neighbor region on the feature in an image.

5.2.6.1 SIFT algorithm

SIFT is an algorithm describe in [2] that extract keypoints and compute its descriptors, as shown in Figure3. In this algorithm there are four mainly steps.



(a) (b)

Figure 2: SIFT extrema Detection

1. **Scale-space extrema detection:** by using Difference of Gaussians, which is an approximation of Laplacian of Gaussian, images are searched for local extrema over scale and space. If it is a local extrema then it is a potential keypoint (Figure 2).

2. **Keypoint Localization:** by adding some thresholds conditions to the potential keypoints found you remove low-contrast keypoints and just remains the strongest interest points. This is the refinement of the results.

3. **Orientation assignment:** the calculation of the gradient magnitude and direction of a neighborhood around the interest points make possible the assignment of orientation to each keypoint and contributes to matching stability.

4. **Keypoint Descriptor:** the neighborhood of each keypoint is represented as a vector which is the keypoint descriptor and make possible the matching in different frames.

5. **Keypoint Matching:** The keypoints are matched by identifying their nearest neighbors.

5.2.6.2 FLANN Base Matcher

FLANN is based in K-d trees which is a data structure for storing a set of points from a k-dimensional space. K-d trees is very good improving run time efficiency and works well for exact nearest neighbor search in low dimensional data, but quickly loses its effectiveness as dimensionality increases. FLANN in an improve version of k-d trees that can deal with large dimensional data making possible to give a better starting point for the ICP that would help in the alignment of the Point Clouds.

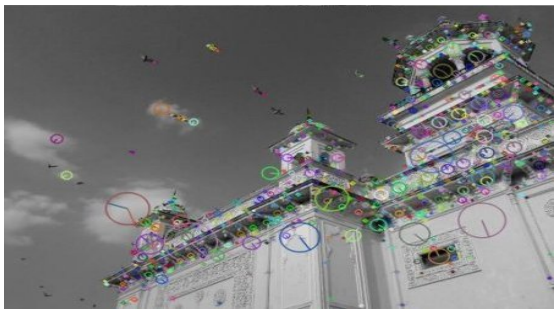


Figure 3: Keypoints of a grey scale image

5.3 Registration

There are two types of registration methods: Rigid and Non Rigid which can be further categorized into pairwise and multiview registration process. They differ mainly in number of pointclouds used for the registration process i.e. pairwise uses two set of pointclouds where as multiview requires multiple sets for minimize the drift accumulated by pairwise registration method.

In brief the most important rigid registration algorithms is discussed below.

5.3.1 Iteration Closest Point -ICP

This is what we have used in our implementation. It is an algorithm used in the process of accurate registration of 3D point cloud data. The objective is to find the transformation that better align the common parts of two point clouds. The algorithm do this refining an initial estimation of the relative transformation of two Point Clouds. Through iterations, the ICP algorithm try to match two points, each one from a point cloud, minimizing the Euclidean distance between them that finally lead to have a better transformation for the next iteration.[3]

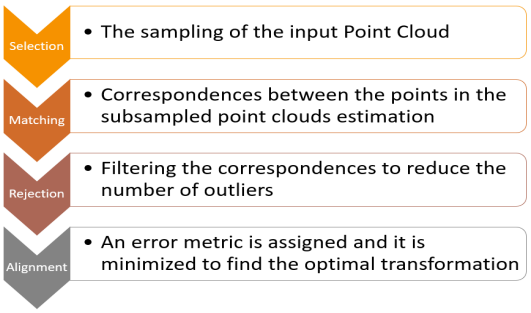


Figure 4: Registration Pipeline

In Figure 4 the general pipeline of the registration process is shown. For every step some consideration are require.

5.3.1.1 Selection

Work with large Point Clouds is computationally expensive so that it is necessary to register only subsets of the Point Cloud or perform downsampling. This is possible considering that there is a lot redundant or unnecessary data and working with less data can yields to the same result and save computation time.

5.3.1.2 Matching

This is the process to match a point in a Point Cloud P with its nearest in a Point Cloud Q in order to find the ideal correspondence. One of the most use strategies to perform matching is using *kd-trees* data structures which use logarithmic searches times.

5.3.1.3 Rejection

This stage consist of filtering the pairs of points matched in the previous stage to facilitate the convergence to the local minimum of the algorithm. This is require because invalid correspondences can affect the registration result and therefore the result. There are different methods to do rejection, some of them, available in PCL are:

- Rejecting pairs with duplicate target matches
- Random Sample Consensus (RANSAC)
- Rejection based on median distance
- Correspondence rejection based on distance
- Rejection based on surface boundaries

5.3.2 Alignment

This is the stage of the error metric minimization. The methods used to do it are:

- Standard point-to-point error metric
- Linear least squares point-to-plane
- Point-to-plane error metric

- Weighted point-to-plane error metric
- Weighted linear least squares point-to-plane
- Generalized error metric and Generalized-ICP

5.4 Meshing

5.5 Point Cloud Data

In PCD, points are represented using X, Y, Z geometric coordinates. In our software, this data has been acquired via Kinect V2 using OpenCV grabber.

The most important part for reconstruction of model data is meshing or commonly known as triangulation. Point data obtained is used to create a detailed object either as implicit function or discrete surface. Three different criterion for accuracy and reliability of the mesh are holes, amount of details and noise.

1. $Holes$: Number of holes and how do they affect the mesh.
2. $Details$: If it is smooth or have information for minute details.
3. $Noise$: Amount of noise in the final mesh which is calculated by number of polygon to be removed explicitly.

Various algorithms works for different kinds of meshes like *Greedy Projection Triangulation* works for smaller meshes with few details whereas $Grid\ Projection$ is applied to larges meshes without much noise. Finally $Poisson$ works where there are no larger holes present and edges are the least.

5.5.1 Greedy Projection Triangulation

It prepares the list of all possible points which could create a mesh. Here, triangulation is obtained by adding edge to the place where there are no edges crossing previously. The only problem is the over representation of the triangles when there are planar points. Where as in non planar points this is fixed using increasing number of boundary points and interpolating between the boundary vertices.

5.5.2 Grid Projection

For surfaces lacking boundaries or proper orientation, it creates a seamless surface unlike above mentioned where there might be a possibility of presence of holes.

5.5.3 Poisson Surface Reconstruction

It combines both the global and local reconstruction method. It create a watertight model by extracting the isosurface and using indicator function. It also works best with noisy data points and hence can retrieve minute details.

6 Improvements

Regarding to the problems faced by the last year project, this project aim to improve the data acquisition and the execution time. The workflow of the project of the project can be seen in Figure 5.

To achieve the proposal goals, the next improvements are implemented:

1. A kinect grabber code were developed in order to adquire depth and color images
2. Used OpenCV library instead of pcl to be able to acquire color and depth.
3. Use SIFT function to initialize ICP
4. Use Modified ICP normal as ICP method instead of ICP general method.
5. Changes in the meshing of the first last year group to get color mesh of the Point Cloud
6. Build a new GUI

The first change come from the acquisition system. Last year projects used a rotation table to give the coordinates and kinect to acquire the images. This project used only the kinect taking into account that this sensor is able to provide coordinates as well as color and depth images.

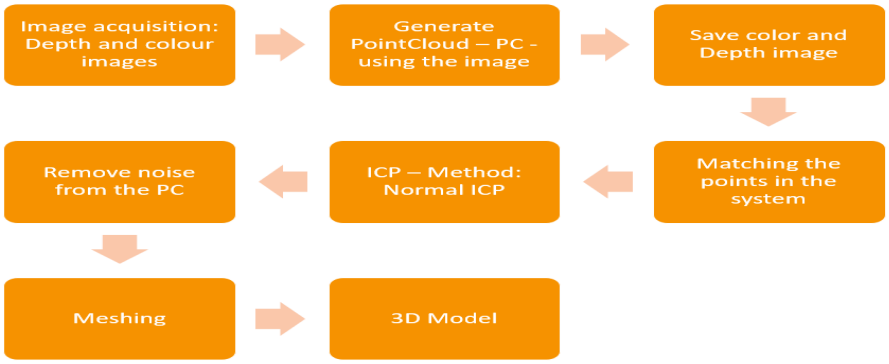


Figure 5: Project Workflow

6.1 Kinect Grabber Code

The kinect grabber code is built with the help of the Microsoft Kinect APIs [4] in order to access the kinect and perform the acquisition. These methods are included in a public classes that were developed by Microsoft to be used with the kinect sensor in windows. The most relevant are:

- ColorFrameSource
- DepthFrameSource
- CoordinateMapper

The basic idea is to set manually the three main parameters: colour frame, depth frame and the coordinate system and then create a dynamic array to display the information acquired. It is important to mention that in order to get the information from the kinect, first we have to link it with the program and check if it is available. We make this possible using *IsOpen* and *IsAvailable* properties. Acquisition is done in various forms like *colormap*, *depthmap* and *color mapped on depth*. After the acquisition we have to remove the outliers from the information to save the processing time and focus on just what is required. To carry out this step we have used the *Threshold* value as explained in the section 5. For our purpose we have used the distance as one meter after which we need to discard the rest of the information using thresholding.

6.2 OpenCV

As in many other applications in computer vision, in this projects a big computational expense is due to the searching for the nearest neighbor matching which mean searching for the most similar matches of vectors. Having a good feature matching algorithm to execute this, can improve the speed and the performance of the software. [5] Is in this task where OpenCV and specially FLANN *Fast Approximate Nearest Neighbor Search Library*- is used.

6.2.1 Feature matching with OpenCV

OpenCV has one major advantage over the feature detection algorithms that were proposed in [6], that is that in OpenCV is possible to get the images from the Kinect and the using libraries and functions that work in image space and that can be more accurate than any other PCL algorithms used, in which the features selected are points that are selected somehow randomly. So, one of the improvements will rely on find better features (that are based in corner detection in images or keypoints detection) and make improvements on software speed.

6.3 Finding Keypoints and Descriptors

In images there are different types of algorithms that are rotation-invariant and help find good features in an image, but when images are rotated and scaled is important to have different approaches to get good features that would be rotation and scaling invariant. With OpenCV there are two major approaches, those are SIFT and SURF.

6.3.1 Scale-Invariant Feature Transform(SIFT)

SIFT algorithm is used in the software to obtain the keypoints and descriptors of every frame. In OpenCV SIFT functionalities [7] are used with **sift.detect()** which finds the keypoints in the images, **sift.compute()** which computes the descriptors of the keypoints found, **sift.detectAndCompute()** that makes the two previous steps in one code line and **cv2.drawKeyPoints()** to graphically represent the keypoints and descriptors over the image (as shown in Figure6).

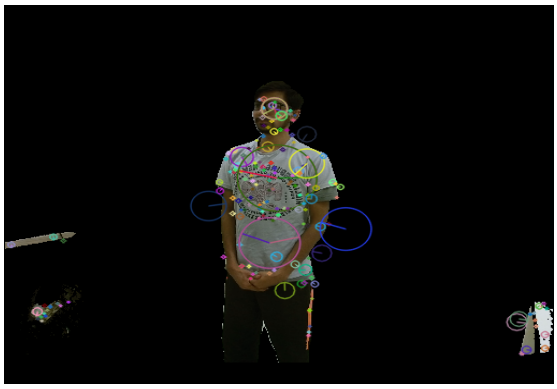


Figure 6: Found keypoints with the software

6.4 Feature matching with FLANN

FLANN stands for Fast Library for Approximate Nearest Neighbors. This matcher contains a collection of algorithms optimized for fast nearest neighbor search [8]. This OpenCV functionally help the software find the closest point between a set of keypoints from two images. For FLANN based matcher two parameters has to be passed. First the dictionary *IndexParams* and the second parameter *SearchParams*. This both dictionaries specify the algorithm to be used and the number of times the trees in the index should be recursively traversed. In Table 2 functions that have been used are shown.

FLANN Functions
cv2.FlannBasedMatcher()
flann.knnMatch()
cv2.drawMatchesKnn()

Table 2: OpenCV FLANN Functions

The results of the Robust match based on the Flann algorithm can be seen in Figure 9 where the closest first 8 important points between two images are shown and kept to use them for find the Affine matrix for the ICP algorithm.

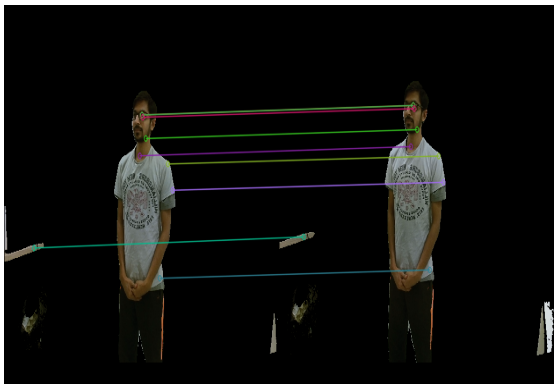


Figure 7: Compute best matches with FLANN Matcher

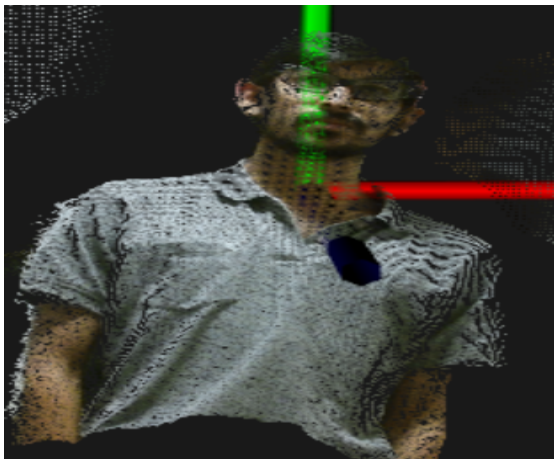


Figure 8: Output from OpenCV images to Point Cloud

6.5 Modified ICP Normal

In the development of this project, a Point-to-Plane alignment criteria called Normal Iterative Closest Point -NICP- method. NICP is a variant of ICP that use point-to-plane distance instead of point-to-point which allows flat region slide each other. As is stated [9] Point-to-plane prove to be more stable and converges faster than the standard method.

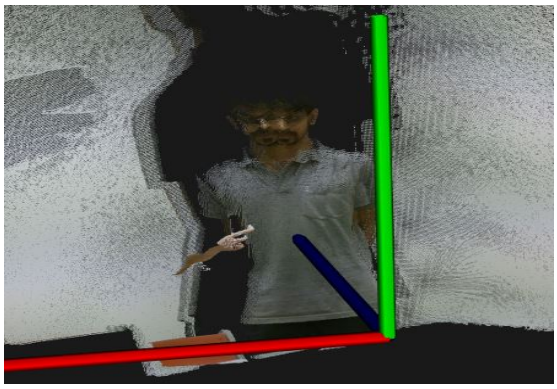


Figure 9: ICP Output

6.6 Meshing

In the first group project of 3D Korn , they created a mesh from the PCD file. So we made modification in the final output of the system. We masked the color frames on the mesh output. As we are masking the color image onto the mesh , it creates a bad color mesh due to the background. We need to add the threshold color image , that can be obtained from the color image if we use openCV library which has been implemented in our scanner code.

6.7 Graphical User Interface

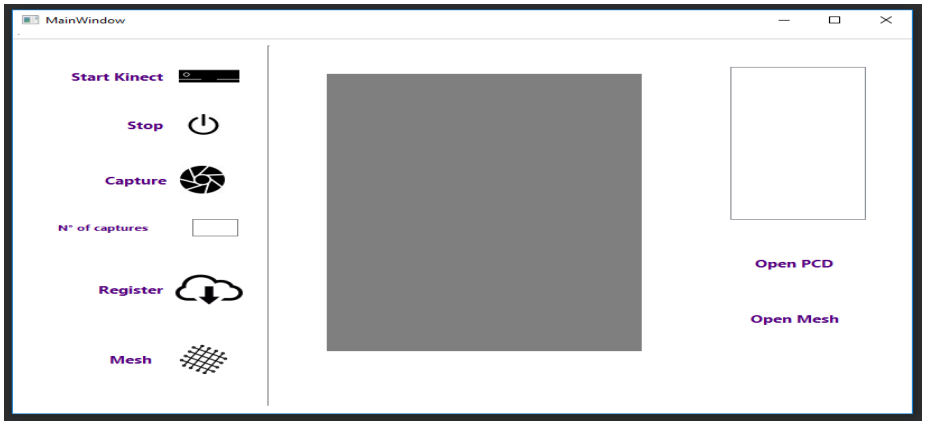


Figure 10: Graphical user interface of 3D Modeling

The GUI implemented in the software is shown in Figure 10. The

basic functions of every button are:

- Start Kinect: Button for start getting images from Kinect
- Stop: Stop the Acquisition from the Kinect Graber
- Capture set: Specifies the number of captures that you want to obtain from the Kinect, acquiring depht and color images.
- Register: Starts the Registration algorithm with the captures obtained
- Mesh: Starts mesh algorithm to show surfaces on the 3D modeling

7 Conclusion

This project gave us a chance to have the broader understanding of many state of the art techniques which can be used for $3D - Reconstruction$ of any object. The liberty to use codes from the previous year's projects became one of the challenge for us to get the whole mindset of the earlier group members. Even though much time was consumed for the installation of softwares and understanding the reverse engineering of project, still we believe to have contributed adequately. Some of which could be mentioned as follows; First of all we have changed the acquisition part from the Kinect and used $OpenCV$ which made the higher performance in terms of feature matching and flooded us with numerous option to play with the images instead point cloud. But to have the $3D-Reconstruction$ we eventually required point cloud, so we have converted the obtained, filtered and smoothened images in the form of point cloud. After this, ICP is required to be done requiring parameters such as r and t which are calculated using $SIFT$ algorithm inbuild in $OpenCV$. Even the general ICP method used in earlier report was replaced with Normal ICP to provide better result. Other contributions from our side could be elaborated as creating the colored mesh instead of the grayscale.

References

[1] Alexander Mordvintsev Abid K. Understanding features. http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_features_meaning/py_features_meaning.html#features-meaning. Accessed January 1, 2018.

[2] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[3] NICP. About normal icp. http://jacoposerafin.com/nicp/index.php/about-ni. Accessed January 4, 2018.

[4] WindowsPreview.Kinect. Kinect sensor class. https://msdn.microsoft.com/en-us/library/windowspreview.kinect.kinectsensor.aspx?cs-save-lang=1&cs-lang=cpp#code-snippet-1. Accessed January 1, 2018.

[5] Andrew W. Moore. An intoductory tutorial on kd-trees, 1991.

[6] Gopikrishna HONG Ziyang LI Chunxia NARAYANA Avinash PULIGANDLA Anirudh STOEVA Darja WAGH Shubham ZAAL HAssan ABUBAKR, AbdelRahman ERABATI. 3d human body scanner. Technical report, University of Burgundy, 2017.

[7] Alexander Mordvintsev Abid K. Understanding features. http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_features_meaning/py_features_meaning.html#features-meaning. Accessed January 3, 2018.

[8] Alexander Mordvintsev Abid K. Feature matching. http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_matcher/py_matcher.html#matcher. Accessed January 4, 2018.

[9] D. Holz, A. E. Ichim, F. Tombari, R. B. Rusu, and S. Behnke. Registration with the point cloud library: A modular framework for aligning in 3-d. *IEEE Robotics Automation Magazine*, 22(4):110–124, Dec 2015.