Project : 3D Scanner

Software Engineering Progress Report : 7

Group Number : 2

Submitted to : Yohan Fougerolle

Dated : 21/11/2016

Centre Universitaire Condorcet - Master Computer Vision

1 Objective

The goal of this report is to briefly explain the development,difficulties and future aspects of the project 3D scanner with a small summary of overall progress.

2 Tasks Assigned Last Week

- Planar Removal
- Performing ICP
- Algorithims for 3D-Reconstruction
- Streaming through GUI with advancements
- Code Optimization

2.1 Planar Removal

- **Algorithm and results**

  First algorithm: As in last week one of our group member worked on Planar removal with quiet satisfying result with one of the algorithm out of two algorithms , but the code was written in visual studio so this week the task was assigned to her to transfer the code from visual studio to qt in order to have everything on one platform and we as group to work on first algorithm instead of second algorithm as we need our point cloud data to be without planar, so we will remove planar before storing it into a vector.

Note:

- A new report will be sent in the upcoming week to update previous report and provide very detail information with code transferred in qt.
- Github link : https://github.com/WajahatAkhtar/Project-S.E/tree/master/Source-code-3D-Scanner/Individual-Operation-Source-Code/Planar-Removal

2.2 Performing ICP

we are working on ICP and trying to understand the algorithm totally, because it's a complex task and an important part of Project which requires good understanding of the algorithm, but after doing some research and reading papers we have been able to get a better idea of how ICP works we are trying now to implement what we have understood and what we were lacking in our past work same as we mentioned in our last week report that we are not getting the desire results we want to, we have tried to use different techniques used before

applying ICP in order to get our desire result such as ,Firstly key estimation , incremental point cloud registration, Normal estimation, correspondence estimation, correspondence rejection and getting a transformation matrix with point of convergence but when we stich point cloud after alignment it gives result for 2 point cloud but start removing some part of the point cloud as we increase our calculations. We have also researched on bundle adjustment in order to stop the point clouds to flip because it will lead us to get our desired result, we have also tried to stitch two point clouds one as source and one target after stitching store there result in another binary or ASCI file such as .pcd or p.ply which will act as input for the next point cloud but we couldn't got the desired as but as mentioned it's a complex task , so we will require little more time to get the final results with that we are working on management of the whole group and assigning task and arranging meetings and report writings as well.More results and more explanation will be added in the next report, as in this week we tried many things but not got our desired results as mentioned above.

Github link : https://github.com/WajahatAkhtar/Project-S.E/tree/master/Source-code-3D-Scanner/Individual-Operation-Source-Code/icp

Problems: We are facing problem in getting better result, as we tried different approaches mentioned above , but looking forward to get better result soon, as its a complex task so need more understanding of the algorithm.

2.3 3D-Reconstruction

Last week, we tried to implement Poison algorithm using the PCL library and some codes we found on Internet and we got a mesh with 'bubbles' and 'distortions'. (see the figure 1).

This week, we focused on the research of Poisson Algorithm in order to be able to find the cause of our problem and implement our findings. The details are as follows:

- We decided to leave Greedy Triangulation method in favour of the Poisson algorithm as our desired algorithm for 3D Reconstruction, since the latter approach always results a water-tight mesh.
- As per Professor Yohan's advice, we decided to work on developing a detailed understanding of the algorithm we were using i.e. Poisson Reconstruction. For this purpose, we read, other than online blogs, the following articles/papers.

  http://www.cs.jhu.edu/ misha/MyPapers/SGP06.pdf

  http://www.drudoo.com/wp-content/uploads/2015/04/Mesh-Reconstruction-Using-The-Point-Cloud-Library.pdf

  http://www.cs.jhu.edu/ misha/ReadingSeminar/Papers/Huang07b.pdf

http://www.cs.jhu.edu/ misha/MyPapers/ToG13.pdf

- Based on knowledge gained from the papers and the advice of Professor Yohan, we found out that the Depth of the Octree parameter, Normals information and the number of samples in the point cloud play a vital role in obtaining quality results. In order to test out these findings, we adopted the following approaches.

- Since we were getting bubbles on our model, we deduced that the problem lied with the inaccurate normal estimation resulting from noisy point cloud. In an attempt to solve this, we applied the Moving Least Square algorithm on the Point Cloud in order to achieve better surface approximation, followed by the Normal Estimation, as opposed to our previous approach of applying Normal estimation, followed by Moving Least Square algorithm. Another thing that we tried was to change the way we are currently doing the normal estimation required to process the Poisson algorithm (on Internet, we see a post in a blog where it's explain that this problem can come from the normal estimation). We change the parameters for the searching of points mainly. It didn't improve the performance of the algorithm.

- We tried to upsample the input mesh during the MLS (Mean Least-Square) Smoothing. We saw in a presentation on the 3D reconstruction

  http://pointclouds.org/blog/tocs/aichim/index.php

  that upsampling a mesh can help to increase a number of points to have fewer holes to fill with the algorithm. We tried two upsampling methods. First, we tried to implement VOXEL GRID upsampling which was described by the author as the best one. We ran the code with it, the program crashes. Next, we tried with another type of upsampling (Sample Local Plane) but it gave us very bad results (see the figure 3). We will decide later if we will keep an upsampling option, but it will depend mainly from if we are able to use the VOXEL upsampling.

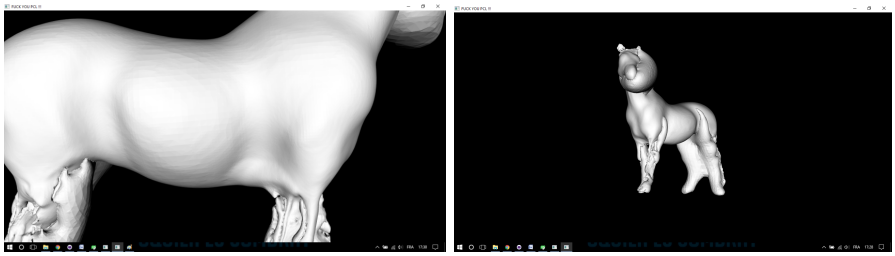

Figure 1: Model with Bubbles

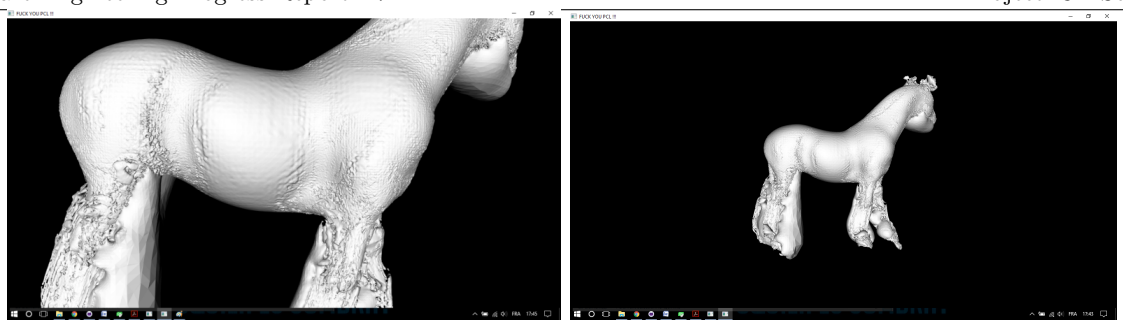

Figure 2: Model with Bubbles

- In the paper

  http://www.cs.jhu.edu/ misha/MyPapers/SGP06.pdf

  we can see the comparison of resolution and time needed, among others. The D parameter is the maximum depth of the octree: it can move from 1 to 14. More you increase the value, more the 3D mesh will be detailed and pronounced but more your program will be slow We used Depth of the Octree = 10, since it provides a good compromise between the resolution and the time needed to process.

Tree Depth	Time	Peak Memory	# of Tris.
7	6	19	21,000
8	26	75	90,244
9	126	155	374,868
10	633	699	1,516,806

Table 1: *The running time (in seconds), the peak memory usage (in megabytes), and the number of triangles in the reconstructed model for the different depth reconstructions of the dragon model. A kernel depth of 6 was used for density estimation.*

Figure 3: Table

Conclusions and Future Work

- We believe that we should do a lot more research and experiments so as to verify our current approach and to discover other solutions. Although we are trying hard to completely understand the enormous theoretical background of the algorithm, we believe that we require some more guidance. To this end, we will be posting our problems on many online forums and also contacting people who have knowledge on this subject .

2.4 Streaming through GUI Qt

GUI Advancement VTK and Kinect Integration

This week, we were able to interface the Kinect with the GUI to perform live streaming of pointclouds. The issue with the crashing interface was due to a call to the function spinOnce() , member of the pcl::visualization::PCLVisualizer class, which was supposed to update the information drawn on the widget. Instead of this, we keep invocating the update () function from the Qvtkwidget, which lets the program to show a constant stream of the pointclouds provided by the Kinect.

The code is arranged in such way that the connection and streaming from the Kinnect are activated by pressing the " Capture " button. Once running, the live pointclouds can be manipulated by using the controls on the left ; for example, if the user moves the slider for Point size , the figure in the screen becomes bolder or lighter. Nonetheless, it's also possible to change other parameters in real time such as the filtering by depth, which lets the user to check what's the best range of acquisition.

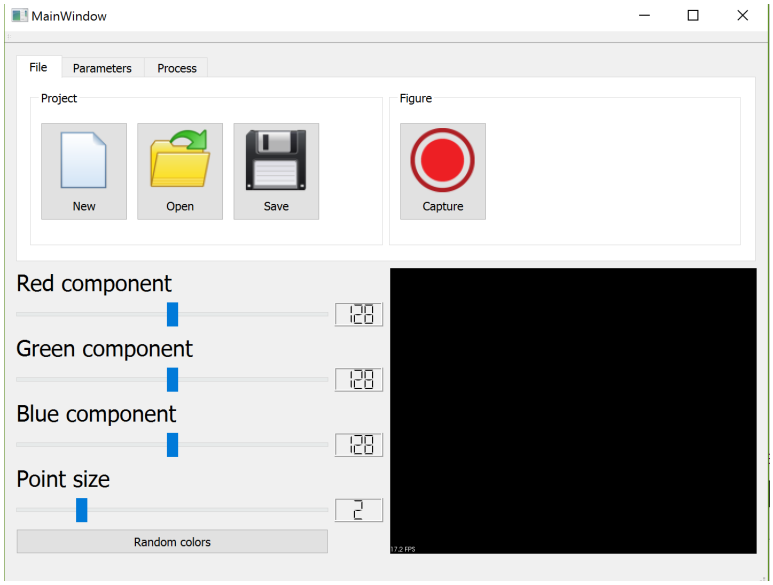

Figure 4: GUI with Streaming Working

The application is also capable of exporting pointclouds into .pcd files every certain time, files to be used later for performing stitching of the pointclouds. Once the algorithms for essential functionalities such as stitching through ICP and 3D reconstruction are succesfully working, the respective classes will be imported and merged into the project.

Regarding the UI design, the group has suggested to rearrange the components to a more Photoshop-like application, with several small windows that will contain all the functionalities, separated and organized according to the stage the user is currently working in.

According to this, next week we will proceed to redesign and implement the new interface, in a way that is easier for the client to understand how 3D scanning works at first sight, bringing also the possibility of choosing only the required information, changing the size of the data, among others:

Github link : https://github.com/WajahatAkhtar/Project-S.E/tree/master/Source-code-3D-Scanner/Un-Optimized-Full-source-Code/With-GUI

2.5 Code Optimization

one of our team has worked on tolérons because we have to embed a clock in grabbing point cloud from kinect v2 and maybe a multithreaded process in our program. As a state of research lot of solutions were found and were able to design a clock in fact the functions will be sent to the group in charge of this to improve the code and to be able to save time. We will try everything to avoid multi thread and if we are able to reduce the process time then we will stop the multi thread idea. If needed to improve the speed we will implement code about multithreaded process to be able to run the livestream in the same time as capturing images and filtering them to store them after in a vector of point cloud.

Github link Optimized Code (same as above) : https://github.com/WajahatAkhtar/Project-S.E/tree/master/Source-code-3D-Scanner/Un-Optimized-Full-source-Code/With-GUI

2.6 Filtering by depth

We were given a task of filtering point cloud data by depth along x and y axis. We tried different values of x,y,z and were able to make a box in which a person can withstand with a window of 1 meter for the movement of that person. We were successful doing that using different values we filtered the image by x-axis (Width), y-axis (Height) and z-axis (Depth). Though we were able to complete our task but we also faced some difficulties also.

Approach Used: We took images after two seconds gap and meanwhile the stream was stopped for that time which made our system very slow, because we were also saving the images in our P.C side by side. Even after deleting the sleep time it was very slow and was not feasible to implement. But when we removed that technique of saving images in P.c, our Code was successfully running on live stream.

Group Discussion: After discussing this problem with my team members. We were given different solution that we will save the point cloud data in a dynamic/resizable vector (accessible to all classes). We have started working on it and soon we will get the results.

3 Future Work

- Appplying Meshes and Finishing ICP
- Enhanced GUI with Streaming
- Optimizing and Combining Code
- Streaming through GUI Qt with advancement
- Studying more algorithm for 3D reconstruction

Github link : https://github.com/WajahatAkhtar/Project-S.E.