Software Engineering

3D Scanner

Submitted by
Meldrick REIMMER
Selma BOUDISSA
Danie SONIZARA
Jaafar Al-TUWAYYIJ

Supervised by **Jiang Cansen and David Strubel**

Contents

List of Figures

Chapter 1

Introduction

For our course module in software engineering we are to review a previous project developed which is based on a 3D scanner. We are to implement this previous project and make any necessary changes should it be. This report is based on findings we found and our understandings, improvements in which we proposed. We work as a group of four and we entirely dedicated our time for this project.

1.1 Background of Study

In order for us to know about the said project we need to educate ourselves on what actually are we going to work on. A 3D scanner is a device that analyzes real-world objects to collect data on its shape and possibly its appearance. The scan data can then be used to construct a digital 3D computer model with measurable properties such as length, width, height, volume, feature size and location, surface area, color, and density. 3d scanning technology.3d scanners come in a wide variety of shapes and sizes, each with unique capabilities and limitations. 3d scanners can be broken down into 3 basic types:

- Contact : Contact 3D scanners use a probe to physically touch points of interest on the objects surface. The scanner is mounted to a fixed surface, while the object it is in contact with rests on a flat surface or is held firmly in place by a fixture. The manufacturing industry uses highly accurate contact scanners known as coordinate measuring machines (CMM).
- Non-Contact Active : Active scanners emit some kind of radiation or light and detect its reflection or radiation passing through object in order to probe an object or environment. Possible types of emissions used include light, ultrasound or x-ray. Time of Flight (TOF)- Resolves the distance to a point based on the speed of light. Light Detection and Ranging (LiDAR) is a common TOF 3d scanner that quickly measures the distance to millions of laser emitted points. Triangulation, Conoscopic, Holography, Structured Light, Modulated Light, Computed Tomography Microtomography
- Non-Contact Passive : Passive 3D imaging solutions do not emit any kind of radiation themselves, but instead rely on detecting reflected ambient radiation (wiki). Photogrammetry, a solution that uses visible light, is an example of a passive 3D imaging. Computers use 2D digital photos to reconstruct an object in 3D by comparing common points taken from different angles. Other types of radiation, such as infrared, could also be used. Stereoscopic, Photometric and Silhouette are types of non-contact passive

A 3D model is a digital representation of a physical object. If you already have an object, and you want it in a digital form, that's what we do. Direct Dimensions takes physical objects that you send to us and we use advanced 3D scanning equipment to capture and transform them into 3D digital models. We do this by processing the raw data gathered during a 3D scan into a digital model that can then be used by you for many purposes. In the next chapter, we'll cover the different methods for collecting this data, including laser scanning and digitizing. A 3D model is incredibly versatile.

3D models can be used for many purposes like making an animation or visualization. They can be used to make design changes to make a new product. They can be used to perform dimensional and comparative analysis of an object, or even FEA and CFD analysis. They can be used for archival purposes - to accurately record the state or form of an object. They can be used to digitally repair damage that has been done to an object which can then reproduce that object in its proper form using rapid prototyping and milling technologies. They can even record your face in intricate detail(And yes, some of our lasers are eye-safe!). There are no limits as to what can be done once something has been captured in 3D.

3D scanners are being applied in several ways. Few examples are:

- Construction and Civil Engineering: The construction and civil engineering sector uses 3d scanners to document changes for final as-built drawings. 3d scan data can also be used to monitor structure deformations and movement.

Figure 1.1: Construction and Civil Engineering

- Quality Assurance and Industrial Metrology: 3d scanners can be used to ensure parts are within tolerance. Manufacturers will use 3d scanners to capture millions of points of interest and compare that to the original CAD data.

Figure 1.2: Quality Assurance and Industrial Metrology

- Entertainment Industry: In a process known as photogrammetry, an ultra realistic animated game teaser for Cyberpunk 2077 was created by 3d scanning actors using an array of high resolution digital cameras. The 3d scans were modified in a modeling and animation application to fit the games theme.

Figure 1.3: Entertainment Industry

- Digital Preservation: 3d scanners can be used to digitally reconstruct fragile objects, such as fossil remains, and to document historical sites. 3d scan data can then be used to analyze complex surface structures without damaging the original object.

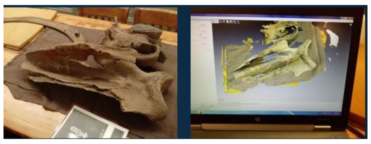


Figure 1.4: Digital Preservation

1.2 Problem Statement

Identify issues of the previous project and work on it, to enable it function better than it was before. The previous project was able to perform a scan of a whole body but had many pity issues especially with the mesh generated and also with the data acquisition.

1.3 Aim

Our aim is to be able to improve the previous project, by proposing and implementing new techniques to build upon it.

1.4 Objectives

To achieve the task we undertook these various objectives:

- Read previous reports of the project.
- Download necessary libraries needed.
- Run the Previous project.
- Locate areas to improve.
- Run tests of improved parts.
- Show results

Chapter 2

Literature Review

There are many existing scanners,from desktop 3D scanners to hand held 3D scanners and advanced metrology systems. Given the low number of reliable 3D scanner reviews, it can be challenging to find the best 3D scanner.

This paper was mainly based on a 3D scanner built by the a group of previous student. They called the Scanner ”3D-Korn”. 3D-korn was developed in 2016. The setup consists of a rotating table and a stationary camera. The person is cap- tured 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. The person is placed on a turning table in front of a stationary depth-sensitive camera and data is collected as the person rotates 360 degrees on the table. We then register the captured data, which is a cloud of points, and use the registered data to create a water tight mesh.

We also drew inspiration from a scanner which was built by two colleagues of our course professor. They were Nicolas Tisserand and Nicolas Burrus. They called their scanner ”Skanect”. Unlike existing technologies, Skanect can acquire dense 3D information about a scene at up to 30 frames per second. Just move around your Structure Sensor, Kinect or Xtion to capture a full set of viewpoints, and you will get a 3D mesh in real time. Skanect makes it easy to 3D scan different kind of scenes by providing a set of predefined scenarios, suitable for most use cases. You can then share your models online in a few clicks - there’s no need to be a trained professional to start 3D scanning Skanect leverages consumer-grade 3D cameras like the Structure Sensor, Microsoft Kinect and Asus Xtion, limiting the hardware cost to a fraction of previous 3D scanning solutions. There are even free versions available in which you can download and use.

Chapter 3

SYSTEM SPECIFICATION AND DESIGN

3.1 Methodology

During system development, there are patterns that needs to be followed in coming up with the complete system. Such patterns includes the waterfall, spiral, RAD, incremental, evolutionary, among others. Our project make use of the V- model which follows a systematic approach in coming out with the complete product. We decided to use this approach because one first gathers and analyses data then from there completes requirements and specification which after sign-off are considered ”set in stone”.

After this stage is the design where the system is designed to come out with a blue print for the implementers who actually perform the coding to follow and this forms the system implementation and testing phase. After implementation which involves deployment, the system is tested to find possible errors to be corrected. After implementation and testing, the system is maintained to ensure that the system is up and running at all times. The waterfall requires a sequential approach to software development that begins with data gathering and analysis and ends with maintenance of the software.

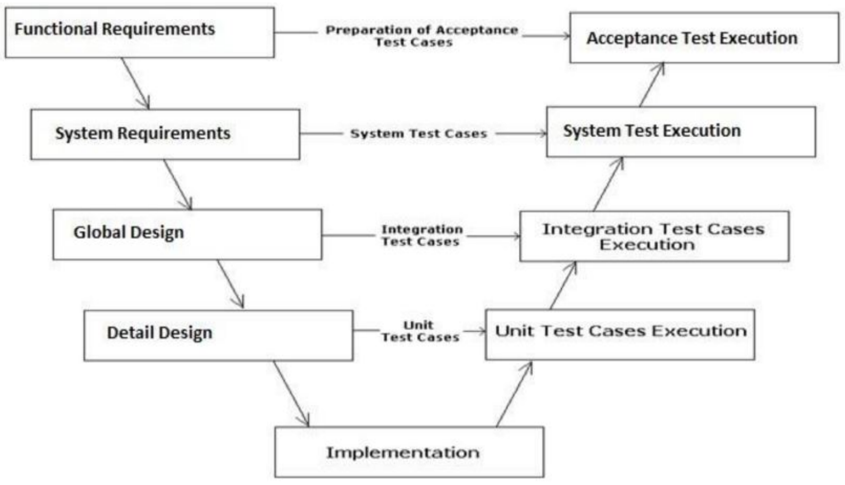


Figure 3.1: V-Model Design

3.2 Functional Requirement

Functional requirements are statements of services the system should provide, how the system should react to particular inputs, and how the system should behave in particular situations. It specifies the software functionality that the developers must build into the product to enable users to accomplish their tasks.

Functional areas are directly related to the various actions that must be taken into consideration and without these characteristics the system cannot meet its required standards. Below are few basic requirements needed.

- The system must be able to scan a whole body.
- The system must be able to acquire data from the scan
- The system must be able to generate point cloud acquisition
- The system must be able to register point cloud data acquired
- The system must be able to perform a 3D reconstruction from the acquire point cloud registered data.
- The system must be able to generate a 3D mesh from the registered data.

3.3 Hardware requirement

3.3.1 Kinect

Description

Kinect was a line of motion sensing input devices by Microsoft for Xbox360 and Xbox One video. Based around a webcam-style add-on peripheral, it enables users to control and interact with their computer without the need for a game controller.

Figure 3.2: Kinect v2

Features

Kinect for Windows V2 offers new features

- Higher resolution capabilites, including three times more depth fidelity and cleaner map
- Wider depth and color field view
- A 25-point skeleton for each of up to six people
- Tracks up to six people simultaneously
- Open-hand and closed-hand gesture recognition
- Biocorrect body joints
- Higher confidence for joints and more points for hands

Specifications

Kinect v2	
Developer	Microsoft
Type	Skeletal tracking
Resolution	1929x1080 (RGB camera)
	512x424 pixels (depth image)
Frequency	30 Hz

3.4 Software Requirement

3.4.1 Qt

Description

Qt is a cross-platform application framework that is used for developing application software that can be run on various software and hardware platforms with little or no change in th underlying codebase.

Specifications

Qt v5.7.0	
Developer	Qt Compagny
Language	C++
Compiler	GCC C++ compiler, Visual studio suite

3.5 Libraries

3.5.1 Point Cloud Libraries

The Point Cloud Library (PCL) is a standalone, large scale, open project for 2D/3D image and point cloud processing.
The PCL framework contains numerous state-of-art algorithms including filtering, feature estimation, surface reconstruction, registration, model fitting and segmentation. These algorithmsq can be used, for example, to filter outliers from noisy data, stitch 3D point clouds together, segment relevant parts of a scene, extract keypoints and compute descriptors to recognize objects in the world bases on their geometric appearance, and create surfaces from point clouds and visualize them.
PCL is a cross-platform, and has been successfully compiled and deployed on Linux, MacOS, Windows and Android. To simplify the development, PCL is split into a series of smaller code libraries, that can be compiled separetely. This modularity is important for distributing PCL on platforms with reduced computational or size constraints.

3.5.2 OpenCV

OpenCV is for Open Source Computer Vision Library is released under a BSD license and hence it's free for both academic and commercial use. It has C++, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android.
OpenCV was designed for computational efficiency and with a strong focus on real-time application. Written in optimized C/C++, the libray can take advantage of multi-core processing.
Enabled with OpenCL, it can tke advantage of the hardware acceleration of the underlying heterogeneous compute platform.
The improvement by using openCV will be on the result for better features matching.

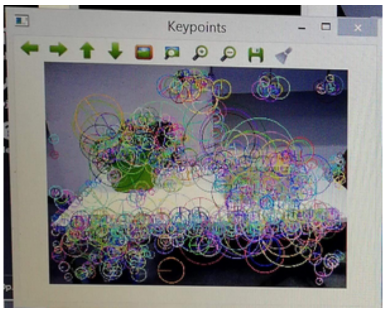


Figure 3.3: Key points

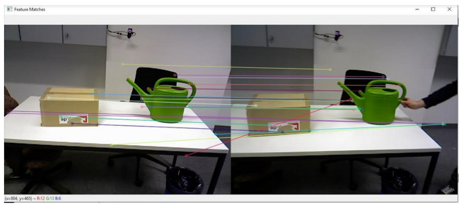


Figure 3.4: Feature matches

3.6 Non Functional Requirements

Non-functional requirements, as the name suggests, are requirements that are not directly concerned with the specific services delivered by the system to its users. They may relate to emergent system properties such as reliability, response time, and store occupancy. Alternatively, they may define constraints on the system implementation such as the capabilities of I/O devices or the data representations used in interfaces with other systems. Non-functional requirements, such as performance, security, or availability, usually specify or constrain characteristics of the system as a whole.

Non-Functional Requirements are the overall qualities or attributes of the resulting system. The following are the non-functional requirements of the System.

- Availability : A system's availability is the amount of time that it is operational and available for use. The system will be ready for use as and when it is needed for transactions
- Reliability and Performance : The system is designed to meet expectations of the user and have a longer performance time without any break down.
- Efficiency : The system will be developed to perform at its and give the right output within the shortest possible time.
- Interface Design: The Graphical User Interface (GUI) is user friendly in order to enhance the interactivity between the user and the system.
- Usability: The system will be designed to give comfort to the user during its usage and making it easier to understand and assimilate

3.7 Use Case Diagram

A use case is a software modeling technique that describes how a system behaves under various conditions as it responds to a request from one of its stakeholders called actors. In essence, a use case diagram tells how an end user interacts with the system under specific set of circumstances. Below is our use case diagram for our system.

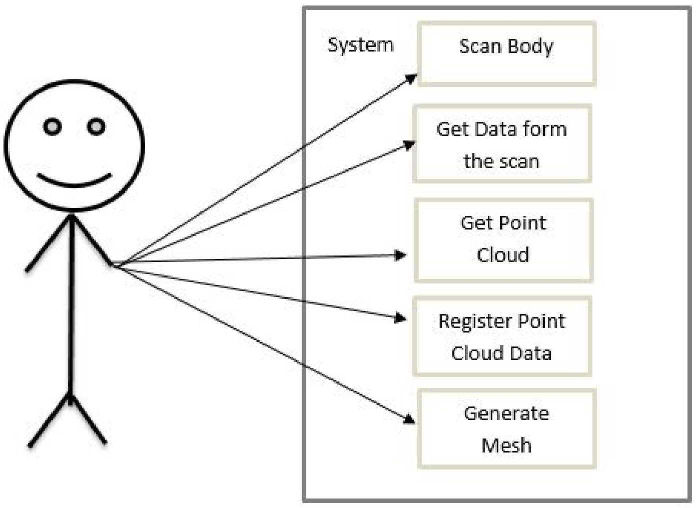


Figure 3.5: Use Case Diagram

3.8 Class diagram

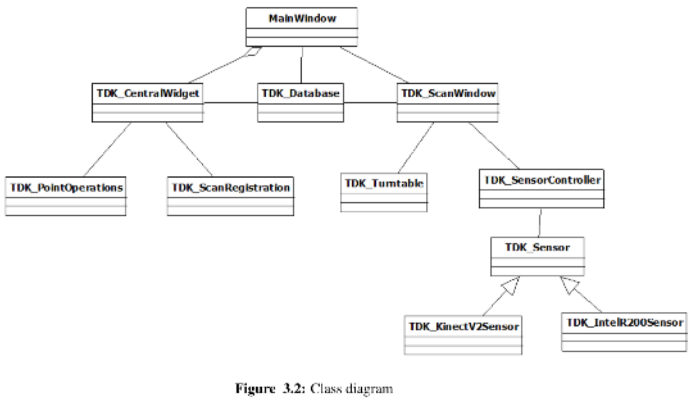


Figure 3.6: Class Diagram of previous project

Chapter 4

SYSTEM IMPLEMENTATION AND DESIGN

System implementation is the deployment of the construction and deployment of the proposed system to start operation and solve the problem it was intended for. The implementation involves putting together, the various designs, modules and database to make it a whole for deployment. After deployment, the system undergoes testing in order to come out with possible errors that need to be corrected.

4.1 Implementation

The previous project was constructed in such manner.:

- Point Cloud Registration
- 3D Reconstruction
- Graphical User Interface (GUI)

4.2 Point CLoud Registration

Kinect captures sequence on snapshots (set of points in three dimensional space, point clouds) of area within the camera frame. Moving Kinect around the stationary object or rotating the object on the turntable in front of Kinect, allows capturing the entire surface of the object. In order to reconstruct the entire surface, individual views must be combined. Each view of the object is slightly transformed compared to previous and for combining the point clouds, relative transformation between two views needs to be found. Once the transformation is found, point clouds can be aligned with each other. The process of finding proper transformation is called point cloud registration. Repeating this process for all the viewpoints allows to align all the point clouds and thus recreate the entire object surface.

There are various algorithms for point cloud registration. They can differ in terms of available input data, if rough estimation of the transformation is known or not, if color information is given. Methods also vary by the kind of correspondences used, motion calculation method, robustness and registration strategy.

Since Kinect provides high frequency snapshots, the relative motion between frames is very small, allowing to roughly estimate the transformation. In such situation, the most commonly used algorithm is Iterative Closest Point (ICP), which finds the registration by minimizing the the distance between point-correspondences, known as closest point.

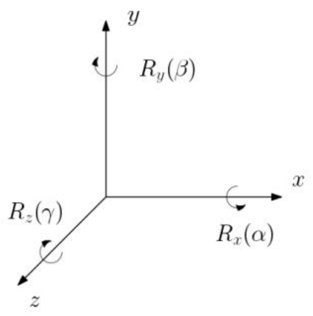


Figure 4.1: Rotation around x-axis, y-axis and z-axis.

4.3 3D Reconstruction

After filming the object the task is to combine consecutive views into single point cloud. For that, first frame is fixed as reference which coordinate system is used. All the other frames are transformed to that coordinate system. Transformation can be calculated iteratively using ICP and alignment process described in 3.3. Initially, first frame is loaded and global transformation matrix Taccum created. In the beginning Taccum = I, where I is identity matrix. For each consecutive frame the following steps are performed

1. Read in depth and color frame and apply Kinect transformation to obtain point cloud of the entire frame. Using Kinect SDK, it is simple to map both color and depth information.
2. Apply pre-processing to extract object point cloud P(current) and point cloud representing data points P(data) for ICP.
3. For point cloud representing model points P(model) use data points from previous frame.
4. Use ICP algorithm to obtain approximated transformation matrix T that describes how P(data) is transformed to P(model).
5. Obtain absolute transformation matrix for current point cloud P(current) by multiplying transformation matrix T from previous point with all other previously obtained transformation matrices T(accum) = T T(accum).
6. Apply cumulative transformation matrix T(accum) to current point cloud P(current) in order to align it to reference coordinate system P(align) = T(accum)P(current)
7. Merge the new aligned point cloud with already aligned points.

Finally, some post-processing is applied on the obtained point cloud.

4.4 Results

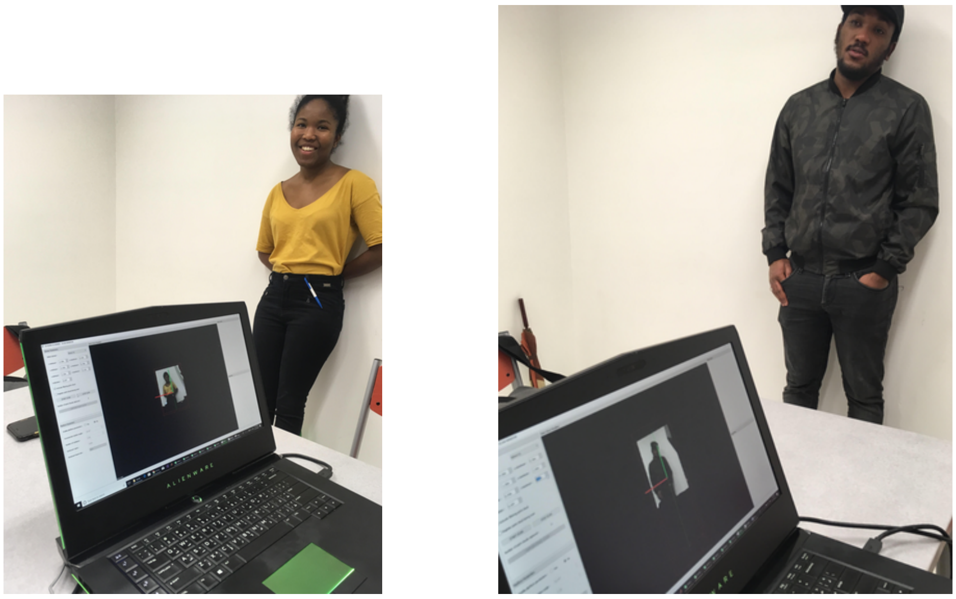


Figure 4.2: Scanning the personns

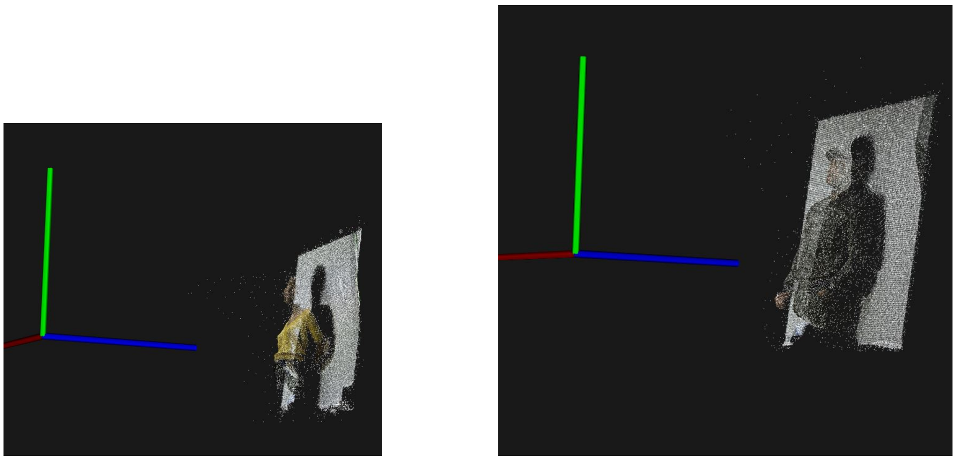


Figure 4.3: Point cloud registration

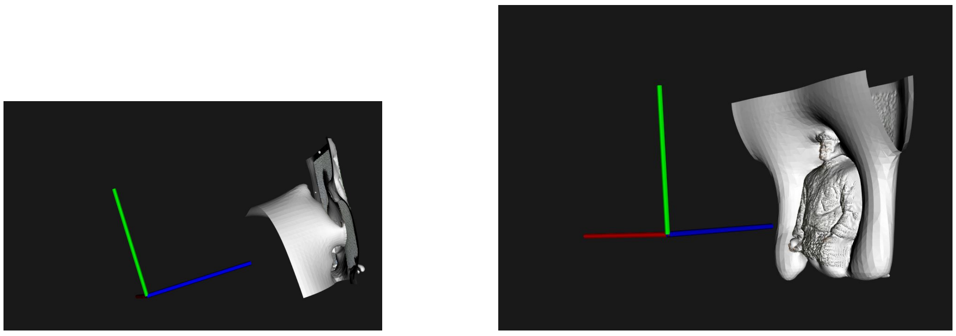


Figure 4.4: 3D reconstruction mesh

Observations: Point clouds obtained from reconstructions are presented in figures above. Since high frequency input data was available, ICP had to register relative small rotations. Unless objects had very specific features, the correct registration could be achieved. Most important factor affecting the quality of the reconstruction was noise and distortions in the input frames. Since Kinect calculates depth based on reflecting light, the reflective properties of object surface severely affect the result. As can be seen from reconstructions, best results are achieved from objects.

Chapter 5

Proposed Improvements

5.1 Graphical User Interface

GUI or Graphical User Interface is a program that takes advantage of the computer's graphics capabilities to make the program easier to use.

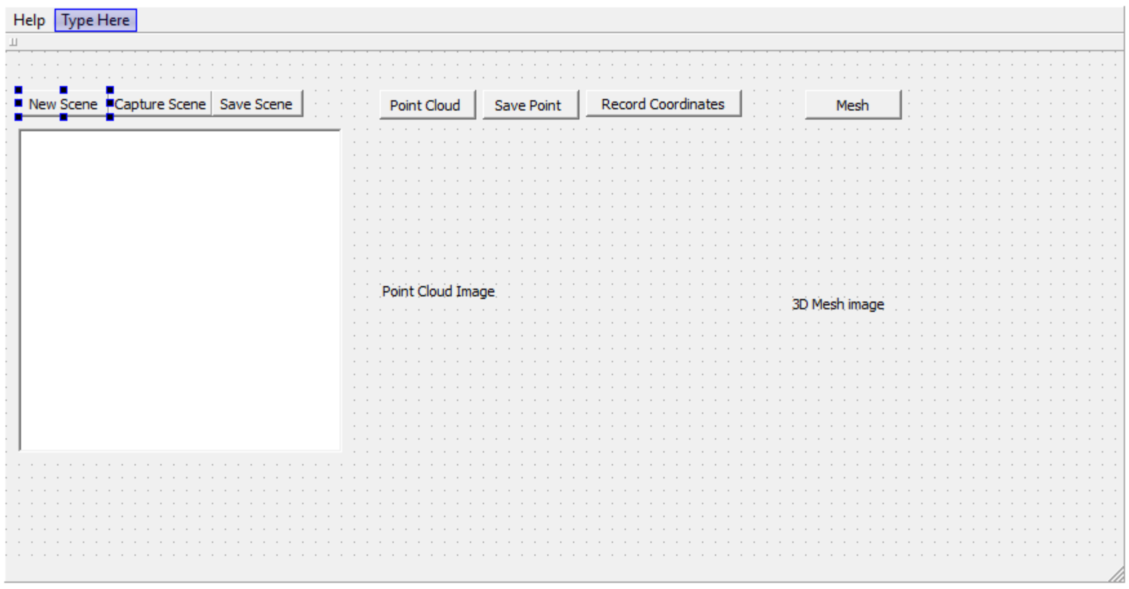


Figure 5.1: Proposed GUI

5.1.1 Scene parameters

New Scene:
The **New Screen** button takes live image recorded by camera and shows it on screen using Qt GUI. So the user can see the scene on live.

Capture scene:
To capture the scene as an image click on the button **Capture scene**.

Save Scene:
To save the image click on the button **Save scene**.Once the button clicked a window file dialog will open so it is possible to rename and save the image in any place in the Desktop.

5.1.2 Viewing the point cloud

Point Cloud:
To get the data base containing points in three-dimensional coordinate system from the captured image, click on the button **Point Cloud**

Save Point:
Every time you click on it,a window file dialog will open so he point cloud images can be saved.

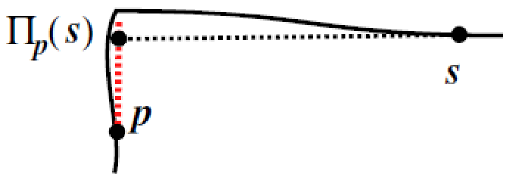
Record coordinates:
This button will record all the data/coordinates from the point cloud images got at different positions

Mesh Finally, to define a 3D shape of the scanned image click on the button **Mesh** .

5.2 Proposed improvement on 3D Mesh

5.2.1 Feature Preservation

The fact that bilateral filtering is feature preserving in the image domain does not, per se, make it feature preserving when extended to 3D. We explore the 3D filter's behavior near corners using the predictors above, with an eye to understanding its treatment of features. We examine a single corner, separating two mostly flat regions. For now, we assume the surface is noise free. If we consider two samples on the surface, s and p,



both on the same side of the feature, then the prediction Î p(s) is likely to be very close to s, as shown in figure 4-2 (regardless of which predictor we use). The term for the influence weight in the filter for this prediction, will therefore be near its maximum. The weight of this prediction will therefore depend for the most part on the spatial weight function.

5.2.2 Normal Estimation and Improvement

The predictors we have discussed require normals (actually, tangent planes) to form predictions. Normals might be provided, or need to be estimated from the data. In either case, they are likely to be noisy, and need some sort of smoothing, or mollification [Huber 1981; Murio 1993], before use in predictors. The need for mollification of noisy normals will be demonstrated and discussed in our final report.

5.2.3 Smoothing Triangle Models

In the context of smoothing triangle models without connectivity, we proposed using a predictor based on facets, but applied to vertex positions [Jones et al. 2003], by taking p as the centroid of the facet. In the limit, this is the same as the approach discussed above for points. As the facet normals are first-order entities of the vertices's, noise in the vertices's is magnified in the normals and predictions. In order to address this, we applied a mollification pass, using a standard blurring filter on the vertex positions to create a smoothed model (without feature preserving). The original vertex positions were not modified in this pass; instead, the normals of the facets from the smoothed model were copied to the corresponding facets of the original, and these smoothed normals were used for prediction. This method was found to be sufficient for almost all models.

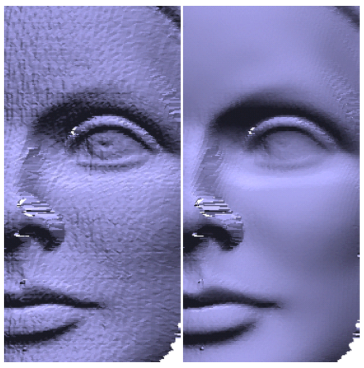


Figure 5.2: Original noisy face scan, and smoothed with the method described

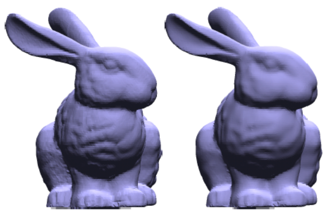


Figure 5.3: Original model, and model smoothed with narrow spatial and influence weight functions

Chapter 6

Conclusion

The goal of this project was to create our own 3D scanner based on the code of the previous year students. They start to implement from scratch and we had to change their code in way to improve it, to get a better reconstruted mesh
We implemented and verified the applicability of a system for fast and efficient 3D reconstruction of objects using the Microsoft Kinect v2, and investigated its suitability, as well as its strengths and weaknesses, in terms of achieving the foregoing goal. First, the depth measuring technology utilized by the Kinect v2 sensor was described, and its theoretical limitations were analyzed.
Next, the proposed method was introduced through a step-by-step analysis of all the modules involved in the system, consisting of the stages starting from image acquisition and proceeding towards formation and postprocessing of a global point cloud representing the object.

References

- **Kinect v2** https://fr.mathworks.com/help/supportpkg/kinectforwindowsruntime/ug/key-features-and-diffe html#bu_d0m7
- **PCL definition** http://pointclouds.org/ http://docs.pointclouds.org/1.8.1/
- **OpenCV** https://opencv.org/
- **OpenCV fetaures matching**
- **Qt 5.7.0** https://en.wikipedia.org/wiki/Qt_(software)

  http://um3d.dc.umich.edu/wp-content/uploads/2015/10/3D-Scanning-Novice.

  https://www.tuit.ut.ee/sites/default/files/tuit/atprog-courses-bakalaureuset55-loti.05.029-lembit-valgma-text-20160520.pdf
- **Previous year code group 1** https://github.com/umaatgithub/3D-KORN/blob/master/Reports/3D-KORN-Software_ Engineering_Project_Final_Report.pdf