Faculty of Engineering and Information Technology

Human Body 3D Scanner (Virtual me)

Esteban Gabriel Andrade Zambrano

Student Number: 12824583

Project Number: AUT-21-04017

Major: Mechanical and Mechatronics Engineering

Supervisor: Dr. Teresa Vidal Calleja

A 12 Credit Point Project submitted in partial fulfilment of the requirement for the Degree of

Bachelor of Engineering

May 25, 2021

Statement of Originality

I, Esteban Gabriel Andrade Zambrano declare that I am the sole author of this report, that I have not used fragments of text from other sources without proper acknowledgment, that theories, results and designs of others that I have incorporated into my report have been appropriately referenced and all sources of assistance have been acknowledged.

Refer to Appendix for signed statement of originality

Abstract

Human Body 3D Scanner (Virtual me) (12cp)
Esteban Gabriel Andrade Zambrano - AUT-21-04017
Supervisor: Dr. Teresa Vidal Calleja
Major: Mechanical and Mechatronics Engineering.
Objective: Process recorded sensor data from a human body and create a scaled model of the scanned subject that can be used for clothing fitting.

The capability of scanning different models is critical for manu industries ranging from fashion to medical and manufacturing. Nevertheless many of the current implementations are extremely expensive. Hence it imposses a barrier for the technology. Similarly, many of the current applications do not create extremely accurate models and in various occasions thgse models would be discarded.
Therefore, this project will create both hardware and software implementation for a human 3D scanner with a moderate budget as well as accurate results in order to be able to produce accurate and scaled models of the scanned subject. The proposed method for data acquisition will be with a custom rig. The rig will be a set of poles with cameras that will capture images from the person that is inside the rig. Additionally, a Lidar will be used to capture a PointCloud as well as an Image mof the subject inside the rig.
With the acquired images from all the cameras as well as the Lidar, a photogrammetry process will be used in order to obtain an initial mesh. This mesh is the result of the photogrametry reconstruction process. Moreover, the initial mesh will be processes with a series of developed algorithms in a pipeline sequence. The pipeline will process the mesh with operations such as plane removal, outliers removal and clustering.
The Scaling will use the PointCloud from the Lidar as a reference and adjust the Scale and Pose of the processed mesh. Once the Mesh has been scaled appropiately with the reference PointCloud, a Poisson algorithm will be used in order to reconstruct and obtain the final scanned model.
The results in this report illustrate the process on how with a constrained data set, it is possible to obtain accurate scaled scanned model that can be used in many applications.

Acknowledgements

I would like to acknowledge and thank my supervisor Dr. Teresa Vidal Calleja for her help and support throughout this project. I would also like to acknowledge and thank Dr. Cedric Le Gentil for all his guidance, support and input throughout the project development. Furthermore, I would like to acknowledge Mr Asher Katz for his contributions in the development of the rig and the image acquisition process component of the project.
On the other hand, I would also like to acknowledge Dr. Mark Liu as well as the UTS Robotics Institute for assitance and resources provided for the completion of this project.

Similarly, I would like to acknowledge the Open Source Community, as assets developed by this community helped in the development and completion of this project.

Finally, I would like to thank my family for their massive and ongoing support throughout this project and my studies. In equally importance I would like to thank God, who with his countless blessings have gave me the opportunity to develop and complete this project.

Contents

List of Figures

List of Tables

Introduction

1.1 Reseach Question

"Human Body 3D Scanner: The development of software for 3D data reconstruction of a Human body scanner with multiple sensors"

1.2 Project Contextualization

The project is based on creating a Human Body 3D scanner. It will have two specific streams that include the development of the mechatronic design of a 3D scanner for a human and the software development for 3D data reconstruction. This proposal is based on developing the software for 3D data modelling and reconstruction of the Scanned data.

Similarly, with the 3D reconstructed model of the human has the aim to be utilised to test different fashion clothing items. This has the intent to adjust the sizing of the clothes fittings based on the Scanned data. The clothing models will adjust automatically depending on the dimensions of the data of the scanned model. The project will have different stages that range from testing different sensors for data acquisition, testing different data stitching frameworks to the deployment of the software in the 3D scanner mechatronic device.

1.3 Problem Definition

Being able to scan different object and models is crucial for many industries. Many applications are in the used in the fashion industry, medical industry, manufacturing ,etc. However, many of the given implementations are extremely expensive , thus making the technology inaccessible for many companies and users in general. There are many forms of implementations, as there are multiple technologies in the market that facilitate the

process in which several devices and software techniques are used. Nevertheless, there is no current industry application that maximises the potential use of the Human body 3D models. Many of the challenges faced is that the software implementation for 3D reconstruction of the models is not particularly accurate, therefore creating imperfect models that on many occasions will need to be discarded.

Hence, this project component will contribute and develop the technology to produce software that will be able to produce accurate models from the gathered data from the sensors. These models will be utilised to try different fashion items and adapt the size fittings accordingly. With the competition of this project many stakeholders, industries and institutions could rely on accurate software that will allow to create a 3D model of a person or object.

1.4 Background

The human society has the world comprehension of the surrounding world through visual perception. This principle allows differentiating distinctive kinds of shapes, objects, colours, textures, and the spatial pose of the surroundings. Based on this information, it is possible to analyse the number of objects in a determined location, object type, object size, object pose in different coordinate frames. Thus, it impacts how as a society we interact with objects or scenes. As a result, it is essential to imitate this perception to acquire real-world data in different formats that include:

- RGB images
- Depth images
- 3D point clouds
- Multispectral images
- Laser readings

All these acquire data can be obtained from a wide variety of commercial or industrial sensors. With this data, it will be possible to use computer processing techniques to model the object or scene (Murcia et al., 2018).

1.5 Applications

In recent years, the use of 3D body scanners has gain importance in several industries. Within the fashion industry, it can aid clothes manufactures to obtain accurate body measurement data of body dimensions. As mentioned by Sturm et al. (2013), this new

technological approach has the potential to alternate the future of the fashion and clothing manufacturing industry.

With the rise of innovation of 3D image reconstruction, the interest to gather precise measurements of humans has raised. Due to the fact, that in the clothing industry is extremely important to create better fittings for different shapes of human bodies. Furthermore, virtual try-on solutions have gain popularity in physical and online retail stores (Spahiu et al., 2014).

On the other side, 3D scanners have gained participation in the medical industry. These systems are described as "non-invasive and low cost", thus making it appealing for epidemiological surveys and clinical uses. (Treleaven and Wells, 2007) The geometrical measurements could be associated with shape, size, volume, and surface area of the body parts. It could aid to be a sustainable approach to screen children and patients with obesity, deformities, or specific anatomic defects. Therefore, it will ease the diagnose process and allow to treat and monitor medical conditions holistically and improve the life quality of patients with non-invasive tests. The table below illustrates the use of a 3D scanner in the medical field with the purpose to identify and monitor various medical conditions. From which the diagnose, treatment and monitor procedures will differ based on the acquired data.

Application	Epidemiology	Diagnosis	Treatment	Monitoring
Measurement				
Size	Anthropometric surveys	Growth defects	Scoliosis	Fitness and diet
Shape	Screening	Abdominal shape	Prosthetics	Obesity
Surface area		Lung volume	Drug dosage	Diabetes
Volume			Burns	
Visualization				
Head		Melanomas	Eating disorders	
Chest			Facial reconstruction	
Whole body			Cosmetic surgery	

Table 1.1: 3D Scanning Applications
(Treleaven and Wells, 2007).

1.6 Overview

As mentioned before, project consists of developing a 3D scanner that is allowed to scan person and be able to use that data in order to be able to virtually fit different clothing. This capstone project mainly focuses on the software and processing of the data for the 3D reconstruction.

The aim of this project is to develop algorithms and an implementation pipeline that will use the acquired image data as well as the reference pointcloud data in order to produce an accurate and appropiately scaled 3D reconstructed model of a person.
This is reconstructed ans scaled model of the scanned person, is intended to be utilised as the model for testing virtual clothe fitting and be able to correctly determine the size of the garments. This technology could have a massive industry in the clothing industry and online retailers as this could solve an issue with the returns policy due to incorrect sizing for different customers groups.

1.6.1 Methodology

To achive the aim, a rig will be used to capture the data. The Rig will consist of a set of poles with multiple cameras. Furthermore a Lidar Intel RealSense L515 will be used to capture another image an the corresponding pointcloud that will be used for reference. Once the rig captures the data (28 images) as well as the lidar data(1 image, 1 pointcloud), the images will be processed with photogrammetry approach with Meshroom. The Processed data from Meshroom will produce a preprocessed reconstruction mesh.

After this the mesh will be processed in the developed algorithmic pipeline and it will be properly scaled with the reference pointcloud from the Lidar. Once the mesh has been processed and scaled, it will be reconstructed appropiatelyand it will generate a final model of the scanned person with the with only the necessary components and appropriate scale.

Once the process is completed the mesh will be saved to disk as a ply file that can be later use in other applications such as clothing fitting.

1.6.2 Structure

This Report will start with a literature review in order to analyse and fully understand the algorithms and techniques that are used in both components. It will help to understand the techniques used in the photogrammetry step and the developed algorithm pipeline.

Following this, details will be presented and illustrated as to how the process is carried out and the results that were obtained from different data sets. It will fully ilustrated how to import the data to Meshroom, what parameters are required to add and modify in order to change the most robust reconstruction. There will be an indetail explanation of what each step executes.

Furthermore, there will be an detailed explanation of the algorithm pipeline for data processing. Here it will be explain what each component is expected to execute as well as how it was created and wrapped into a framework.

Literature Review

Being able to scan different objects and subjects has been a challenging task for researchers. Getting an accurate spatial location of the objects is crucial for this type of application. The use of 3D point clouds has facilitated this process as it allowed to obtain the following parameters:

- Depth
- Intensity
- Pulse width
- Light echo

This information can be obtained with different kind of sensors. There is a wide variety of off the shelf sensors that can provide 3D point clouds. These sensors could either be stereo or multiview vision cameras, lasers, time-of-flight sensors (TOF) and structured light sensors as stated by Murcia et al. (2018).

Many Scanning devices will use single or multiple of the above-mentioned sensors to acquire data. Once the data is obtained, it essential to have a framework for 3D data modelling and reconstruction. The principle behind 3D data reconstruction is obtained with data fusion from RGB-D sensors. This kind of sensors provide 3 channels images RGB (red, green, blue) and the depth images are mapped to each pixel. Based on this data, 3D point clouds could be generated for data reconstruction.

Similarly many other scanning devices use photogrammetry as a technique in order to do the reconstruction. In this particular project Meshroom was used as a 3D photogrammetry reconstruction software.

Moreover, several open source libraries were used in order to process and scale the draft mesh. These open source libraries include PCL and Open3D. From these several algorithms were used that aid in the data processing and these will be explained below.

2.1 Photogrammetry

Photogrammetry is decribed as the associated techniques with performing measurements of real-world objects and terrain features from images as mentioned by Aber et al. (2019). Many of the applications include the quantification of distances, volumes,areas, heights, 3D topographic mapping, measuring of objects, extracion of 3D pointcloud for surfaces reconstruction as well as the generation of orthophotographs and digital elevation models.(Aber et al., 2019).

In recent years, with the development of technologies pairing computer vision concepts & algorithms and photogrammetry specifically *Structure from Motion-Multiview Stereo (**SfM-MVS**)* has led to significant advances in 3D surface Reconstruction from images (Aber et al., 2019).

The key principle behind all the photogrammetric measurements is associated with the mathematical & geometrical reconstruction of the path of light rays from the object to the sensor camera in the exact moment of data image acquisition. Thus, the fundamental concept of photogrammetry is the undertanding of geometric characteristics of a single photograph.

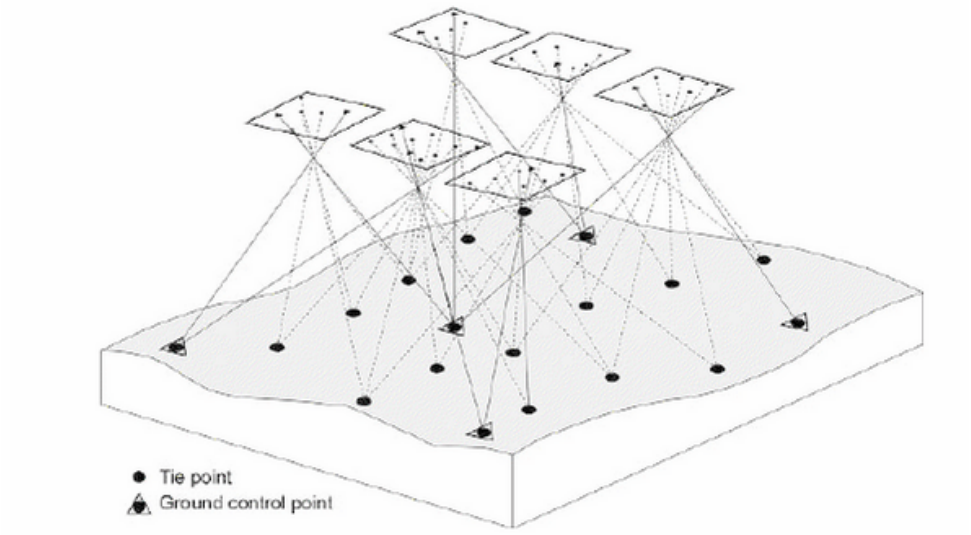


Figure 2.1: Photogrammetry Principle Illustration
(Aber et al., 2019)

2.2 Meshroom

Meshroom is a free, open-source 3D Reconstruction Software based on the AliceVision framework (AliceVision, 2021). AliceVision is a software framework based on photogrametric computer vision. which focuses on the development of 3D Reconstruction and Camera Tracking Algorithms. It aims to provide robust software foundation with novel computer vision algorithms that can be analysed, tested and reused.
Meshwoom was developed as collobaration project between industry and academia in order to develop cutting-edge algorithms that are robust and of high quality which are crucial for production environments.

Components Pipeline

Meshroom can be downloaded and used in both Windows and Linux and it will require a powerfull CPU in order to have adequate performance. Also it will require an Nvidia GPU as it requires to run CUDA for different nodes and components. Once meshroom is downloaded it can run in either platform. When it is executed the Meshroom GUI will appear and it will be similar to figure 2.2

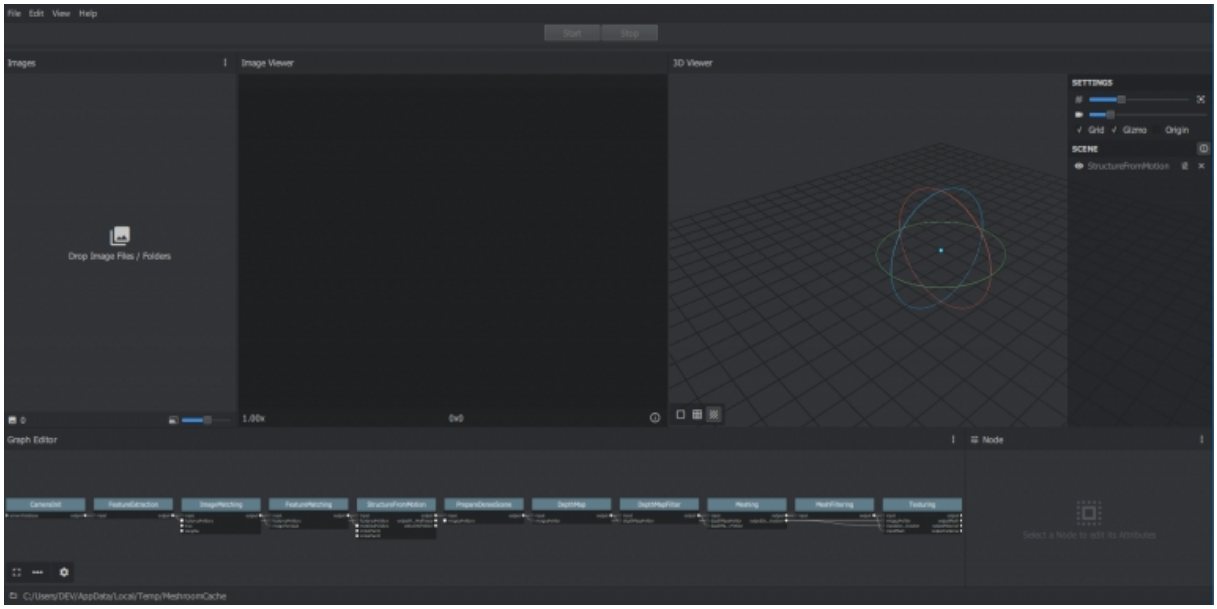


Figure 2.2: Meshroom Graphical User Interface
(AliceVision, 2021)

As mentioned before, meshroom is based on AliceVision's Framework. Therfore it follows a photogrametric pipeline for 3D reconstruction. This pipeline will have the following components:

- Natural Feature Extraction
- Image Matching
- Features Matching
- Structure from Motion
- Depth maps estimation
- Meshing
- Texturing
- Localization

The pipeline will be included into Meshroom GUI and it will be used in the nodes. All the components in Meshroom will be embedded into different individual nodes that will be executed and produce individual results that will be send to the next node along in the pipeline. As the nodes represent specific components of the reconstruction pipeline, these should be executed in a defined order.
The nodes in the defined order to reconstruct a 3D model from images include:

1. Camera Initialization
2. Feature Extraction
3. Image Matching
4. Feature Matching
5. Structure From Motion
6. Prepare Dense Scene
7. Depth Map
8. Depth Map Filter
9. Meshing
10. Mesh Filtering
11. Texturing

2.2.1 Camera Initialization

Camera Initialization or *"CameraInit"* loads the image metadata, sensor information, camera parameters and it will generate viewpoints based on the images. It is possible to use a mixture of cameras & focal lengths. This node will generate a groups of intrinsics that are based on the images metadata, these grooup os intrinsics can be adjusted if the cameras have been fully calibrated. The intrinsic include the camera matrix which includes the focal length (f_x,f_y) and principal point (x_0,y_0).

$$CameraIntrinsics = \begin{pmatrix} f_x & 0 & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{pmatrix}$$

The components in the node are described on table 2.1

Viewpoints Input	viewpoints (1 Element for each loaded image) - ID - Pose ID - Image Path - Intrinsic: Internal Camera Parameters (Intrinsic ID) - Rig (-1 - 200) - Rig Sub-Pose: Rig Sub-Pose Parameters (-1 - 200) - Image Metadata: (list of metadata elements)
Intrinsic Camera Intrinsics	(1 Element for each loaded image) - ID - Initial Focal Length: Initial Guess on the Focal Length - Focal Length: Known/Calibrated Focal Length - Camera Type: pinhole', 'radial1', 'radial3', 'brown', 'fisheye4' - #Make: Camera Make (not included in this build, commented out) - #Model: Camera Model - #Sensor Width: Camera Sensor Width - Width: Image - Width (0-10000) - Height: Image Height (0-10000) - Serial Number: Device Serial Number (camera and lens combined) - Principal Point: X (0-10000) Y(0-10000) - DistortionParams: Distortion Parameters - Locked(True/False): If the camera has been calibrated, the internal camera parameters (intrinsics) can be locked. It should improve robustness and speedup the reconstruction.
Sensor Database	Camera sensor width database path
Default Field Of View	Empirical value for the field of view in degree 45° (0°-180°)
Verbose Level	verbosity level (fatal, error, warning, info, debug, trace)
Output SfMData File	.../cameraInit.sfm

Table 2.1: Meshroom CameraInit Node

2.2.2 Feature Extraction

The key objective of this node is to extract distinctive pixel groups, which are to a certain extend invariant to a change in camera viewpoints during the image acquisition process. Thus, a feature in this scene should have similar features descriptors in all the captures images.

One of the most used feature detection method is **SIFT** *(Scale-invariant feature transform)* algorithm. The objective of SIFT is to extract discriminative patches in an initial image that later can be compared to discriminative patches of a second image, irrespective of scale, rotation and translation (Lowe, 2004). As a relevant detail is only present at a specific scale, the extracted patches are centered around the point of interests. Therefore,

the SIFT invarience can be used to deal with image transformations that occur when the viewpoints change during image acquisition.

Based on the representation of one image at different scales, SIFT is able to compute scale-space maxima of the Laplacian Representation. This is a defined image energy-based representation, using the differences of Gaussians. This Maxima is associated to the points of interest in the image. Once this is processed, it samples for each of the this maxima a square image patch, whose origin is the maximum and "x" direction is the dominant gradient at the origin as suggested by Lowe (2004) andOtero (2015) . For each keypoint, a description of these patches is associated.

The description consists of statistics of gradients which is computed in regions around the keypoint. The size of the region is defined by the keypoint scale and the orientation by the dominant axis. It is also frequently stored in 128 bits.

The number of extracted features could vary as a result of texture complexity, from one image to other ones or in different image sections. Hence, a post-filtering step controls the number of extrated features to a specified limit. Furthermore, grid filtering is used to ensure an even repartition in the image.

The components in the node are described on table 2.2

Name	Description
Input	SfMData file.
Describer Types	Describer types used to describe an image. 'sift', 'sift*float', 'sift*upright', 'akaze', 'akaze*liop', 'akaze*mldb', 'cctag3', 'cctag4', 'sift*ocv', 'akaze*ocv'
Describer Preset	Control the ImageDescriber configuration (low, medium, normal, high, ultra). Configuration "ultra" can take long time !
Force CPU Extraction	Use only CPU feature extraction.
Max Nb Threads	Specifies the maximum number of threads to run simultaneously (0 for automatic mode). (0-24) 0
Verbose Level	verbosity level (fatal, error, warning, info, debug, trace).
Output Folder	Output path for the features and descriptors files (*.feat, *.desc).

Table 2.2: Meshroom FeatureExtraction Node Settings

2.2.3 Image Matching

The principle behind this node is to find images that point to the same areas of interest. Therefore, image retrieval techniques are implemented with the purpose of finding images that share content without the demand of resolving all the detailed feature matches. Hence, the goal is to simplify the image in a compact image descriptor. This allows to efficiencly compute the distance between all images descriptors.

A vocabulary tree is one of the widely used methodologies to generate the image descriptor. It works by passing all the extracted features descriptors into it. Then it performs a classification process, which compares their descriptors to the ones on each node of the vocabulary tree as mentioned by Nistér and Stewénius (2006). Each feature descriptor is associated with one leaf, which can be stored with an standard index (*The index of*

this leaf in the tree).Thus, the collection of leaves indices represents the image descriptor (Nistér and Stewénius, 2006).

The components in the node are described on table 2.3

Name	Description
Image	SfMData file
Features Folders	Folder(s) containing the extracted features and descriptors
Tree	Input name for the vocabulary tree file ALICEVISION_VOCTREE
Weights	Input name for the weight file, if not provided the weights will be computed on the database built with the provided set
Minimal Number of Images	Minimal number of images to use the vocabulary tree. If we have less features than this threshold, we will compute all matching combinations
Max Descriptors	Limit the number of descriptors you load per image. Zero means no limit
Nb Matches	The number of matches to retrieve for each image (If 0 it will retrieve all the matches) 50 (0-1000)
Verbose Level	verbosity level (fatal, error, warning, info, debug, trace)
Output List File	Filepath to the output file with the list of selected image pairs

Table 2.3: Meshroom Image Matching Node

2.2.4 Feature Matching

The key component of feature matching node is to be able to match all features between potential image pairs. Initially, the node performs a photometric process that creates matches between the descriptors from 2 separe images. For each feature in image "A", a list of potential features in image "B" is generated. The descriptor space is not linear and defined space, which causes uncertainty in the validity of the matches due to the absolute distance values. Furthermore, in order to remove bad candidates, an assumption process associates only one valid match in the other image. Hence, for each feature descriptor on Image "A", two of the closests descriptors in the image are used with a relative threshold between them which provides a robust criteria as mentioned by Lowe (2004).

This process will provide a photometric list of feature matching candidates. Afterwards, the images features positions are used in a geometrica filtering process that uses epipolar geometry in an outlier detection framework, which is RANSAC (*"Random Sample Consensus"*). Moreover, a random selection process uses a small set of feature correspondances and it computers either the fundamental or essential matrix. Once this process is completed, the number of features verifies and validates this model and iterates through the RANSAC framework (Muja and Lowe, 2009). The components in the node are described on table 2.4

Name	Description
Input	SfMData file
Features Folder	
Features Folders	Folder(s) containing the extracted features and descriptors
Image Pairs List	Path to a file which contains the list of image pairs to match
Describer Types	Describer types used to describe an image **sift**'/ 'sift_float'/ 'sift_upright'/ 'akaze'/ 'akaze_liop'/ 'akaze_mldb'/ 'cctag3'/ 'cctag4'/ 'sift_ocv'/ 'akaze_ocv
Photometric Matching Method	For Scalar based regions descriptor * BRUTE_FORCE_L2: L2 BruteForce matching * ANN_L2: L2 Approximate Nearest Neighbor matching * CASCADE_HASHING_L2: L2 Cascade Hashing matching * FAST_CASCADE_HASHING_L2: L2 Cascade Hashing with precomputed hashed regions (faster than CASCADE_HASHING_L2 but use more memory) For Binary based descriptor * BRUTE_FORCE_HAMMING: BruteForce Hamming matching
Geometric Estimator	Geometric estimator: (acransac: A-Contrario Ransac // loransac: LO-Ransac (only available for fundamental_matrix model)
Geometric Filter Type	Geometric validation method to filter features matches: **fundamental_matrix** // essential_matrix // homography_matrix /// homography_growing // no_filtering'
Distance Ratio	Distance ratio to discard non meaningful matches 0.8 (0.0 - 1)
Max Iteration	Maximum number of iterations allowed in ransac step 2048 (1 - 20000)
Max Matches	Maximum number of matches to keep (0 - 10000)
Save Putative Matches	putative matches (True/False)
Guided Matching	the found model to improve the pairwise correspondences (True/False)
Export Debug Files	debug files (svg/ dot) (True/False)
Verbose Level	verbosity level (fatal, error, warning, info, debug, trace)
Output Folder	Path to a folder in which computed matches will be stored

Table 2.4: Meshroom Feature Matching Node

2.2.5 Structure From Motion

The Structure From Motion Node carries out the 3D points reconstruction freom the input images. The key behind this node is to be able to comprenhend the geometric relationship behind all the views from the input images, and decifer the rigid scene structure (3D PointCloud) along with the pose (position & orientation) and internal calibration of the cameras as mentioned by Cheng et al. (2014) This is an incremental process pipeline which associated with a growing reconstruction process. It initially computes an starting two-view point reconstruction which is iteratively extended by adding new views (Cheng et al., 2014).

It starts by fusing all feature matches between all images and store them in tracks. Each track represents a point in space, which is visible from multiple cameras. Nevertheless, at this stage of the pipeline, many outliers are present,therefore during matches fusion, incorrect tracks are removed (Fischler and Bolles, 1981).

Afterwards, the incremental algorithm chooses the best initial image pair. This choice is very important for the quality of the final reconstruction, hence it should provide robust matches and reliable geometric information as suggested by Moulon et al. (2012). Therefore, this image pair maximises the number of matches and the repartition of features correspondances in all images. On the other hand, the angle between the cameras must be able to provide with reliable gemoetric information. Then, it computes the fundamental matrix between these two images considering the first image as the origin of the coordinate system (Kneip et al., 2011). As the pose of the first two cameras is known, it is possible to triangulate the matching 2D features into 3D points.

Once this step finishes, the *Next Best Views Selection* algorithm selects all the images that have enough number of associations with the 3D reconstructed features. Based on 2D-3D associations, it performs the resectioning of each of the new cameras (Lepetit et al., 2008).

The resectioning is a **Perspective-n-Point** algorithm (***PnP***) in a **RANSAC** framework that finds the pose of the camera that validates the features association. Similarity, a non-linear minimization proceess is performed on each camera, to refine the end pose (Nister, 2004).

From these new camera poses, some tracks become visible by two or more resected cameras and it triangulates them. A Bundle Adjustment process starts and refines everything such as: intrinsic and extrinsic parameters of all cameras, as well as the position of all 3D points as suggested by Shah et al. (2014). A filter processes the result of the Bundle Adjustment process and removes all osbservations that have a high reprojection error or not enough angles between viewpoints.

In the new points tringulation process, it uses more image candidates for the best views selection. It iterates by adding cameras and tringulatins new 2D features into 3D points and removes 3D points that become invalid in the process,until no more new views can be localized (Shah et al., 2014).

The components in the node are described on table 2.5

Input	**SfMData file**
Features Folder	Folder(s) containing the extracted features and descriptors.
Matches Folders	Folder(s) in which computed matches are stored.
Describer Types	Describer types used to describe an image. 'sift', 'sift*float', 'sift*upright', 'akaze', 'akaze*liop', 'akaze*mldb', 'cctag3', 'cctag4', **'siftocv', 'akazeocv'
Localizer Estimator	Estimator type used to localize cameras (acransac, ransac, lsmeds, loransac, maxconsensus).
Observation Constraint	Observation constraint mode used in the optimization: Basic: Use standard reprojection error in pixel coordinates, Scale: Use reprojection error in pixel coordinates but relative to the feature scale
Localizer Max Ransac Iterations	Maximum number of iterations allowed in ransac step. (1-20000) 4096
Localizer Max Ransac Error	Maximum error (in pixels) allowed for camera localization (resectioning). If set to 0, it will select a threshold according to the localizer estimator used (if ACRansac, it will analyze the input data to select the optimal value). (0.0-100-0) 0.0
Lock Scene Previously Reconstructed	This option is useful for SfM augmentation. Lock previously reconstructed poses and intrinsics.
Local Bundle Adjustment	It reduces the reconstruction time, especially for large datasets (500+ images) by avoiding computation of the Bundle Adjustment on areas that are not changing.
LocalBA Graph Distance	Graph-distance limit to define the Active region in the Local Bundle Adjustment strategy. (2-10) 1
Maximum Number of Matches	Maximum number of matches per image pair (and per feature type). This can be useful to have a quick reconstruction overview. 0 means no limit. (0-50000) 1
Minimum Number of Matches	Minimum number of matches per image pair (and per feature type). This can be useful to have a meaningful reconstruction with accurate keypoints. 0 means no limit. (0-50000) 1
Min Input Track Length	Minimum track length in input of SfM (2-10)
Min Observation For Triangulation	Minimum number of observations to triangulate a point. Set it to 3 (or more) reduces drastically the noise in the point cloud, but the number of final poses is a little bit reduced (from 1.5% to 11% on the tested datasets). (2-10)
Min Angle For Triangulation	Minimum angle for triangulation. (0.1-10) 3.0
Min Angle For Landmark	Minimum angle for landmark. (0.1-10) 2.0
Max Reprojection Error	Maximum reprojection error. (0.1-10) 4.0
Min Angle Initial Pair	Minimum angle for the initial pair. (0.1-10) 5.0
Max Angle Initial Pair	Maximum angle for the initial pair. (0.1-60) 40.0
Use Only Matches From Input Folder	Use only matches from the input matchesFolder parameter. Matches folders previously added to the SfMData file will be ignored.
Use Rig Constraint	Enable/Disable rig constraint.
Force Lock of All Intrinsic Camera Parameters.	Force to keep constant all the intrinsics parameters of the cameras (focal length, principal point, distortion if any) during the reconstruction. This may be helpful if the input cameras are already fully calibrated.
Filter Track Forks	Enable/Disable the track forks removal. A track contains a fork when incoherent matches lead to multiple features in the same image for a single track.
Initial Pair A	Filename of the first image (without path).
Initial Pair B	Filename of the second image (without path).
Inter File Extension	Extension of the intermediate file export. ('.abc', '.ply')
Verbose Level	Verbosity level (fatal, error, warning, info, debug, trace).
Output SfMData File	Path to the output sfmdata file (sfm.abc)
Output SfMData File	Path to the output sfmdata file with cameras (views and poses). (cameras.sfm)
Output Folder	Folder for intermediate reconstruction files and additional reconstruction information files.

Table 2.5: Meshroom StructureFromMotion Node

2.2.6 Prepare Dense Scene

This node undistorts the images and it genera etes EXR images. The components in the node are described on table 2.6

Name	Description
Input	SfMData file
ImagesFolders	Use images from specific folder(s). Filename should be the same or the image uid.
Output File Type	Output file type for the undistorted images. (jpg, png, tif, exr)
Save Metadata	Save projections and intrinsics information in images metadata (only for .exr images).
Save Matrices Text Files	Save projections and intrinsics information in text files.
Correct images exposure	Apply a correction on images Exposure Value
Verbose Level	[‘fatal’, ‘error’, ‘warning’, ‘info’, ‘debug’, ‘trace’]
Output	MVS Configuration file (desc.Node.internalFolder + ‘mvs.ini)
Undistorted images	List of undistorted images.

Table 2.6: Meshroom Prepare Dense Scene Node

2.2.7 Depth Map

This step in the pipeline retrieves the depth value of each pixel for all cameras that were processes by the StructureFromMotion Node. It uses Semi-Global Matching method as proposed by Hirschmuller (2005).

For each image, it selects the **"N"** best & closests cameras around. It chooses fronto-parallel planes based on the intersection of the optical axis with the pixels of the choosen neighboring cameras (Hirschmuller, 2005). This produces a volume **W,H,Z** with several depth candidates per pixel and it estimates the similarity for all of these. The Similarity is calculated by the *Zero Mean Normalized Cross-Correlation (ZNCC)* of a small patch in the principal image which is reprojected into at the other camera, which creates a volume of similarities as suggested by Strecha et al. (2006). For each neighboring image, it accumulates similarities in this volume. Nevertheless, this volume is noisy,therefore, there is a filter in each step along X & Y axis which reduces the isolated high values. Finally, it selects the local minima and replaces the selected plane index with a depth value stored into a depth map (Scharstein et al., 2001). This depth map has banding artifacts as it based on the original selection of depth values as mentioned by Scharstein et al. (2001). Thus, a refinement process calculates the depth values with sub-pixel accuracy.

The components in the node are described on table 2.7

Name	Description
MVS Configuration File:	SfMData file.
Images Folder	Use images from a specific folder instead of those specify in the SfMData file. Filename should be the image uid.
Downscale	Image downscale factor (1, 2, 4, 8, 16)
Min View Angle	Minimum angle between two views.(0.0 - 10.0)
Max View Angle	Maximum angle between two views. (10.0 - 120.0)
SGM: Nb Neighbour Cameras	Semi Global Matching: Number of neighbour cameras (1 - 100)
SGM: WSH: Semi Global Matching	Half-size of the patch used to compute the similarity (1 - 20)
SGM: GammaC	Semi Global Matching: GammaC Threshold (0 - 30)
SGM: GammaP	Semi Global Matching: GammaP Threshold (0 - 30)
Refine: Number of samples	(1 - 500)
Refine: Number of Depths	(1 - 100)
Refine: Number of Iterations	(1 - 500)
Refine: Nb Neighbour Cameras	Refine: Number of neighbour cameras. (1 - 20)
Refine: WSH	Refine: Half-size of the patch used to compute the similarity. (1 - 20)
Refine: Sigma	Refine: Sigma Threshold (0 - 30)
Refine: GammaC	Refine: GammaC Threshold. (0 - 30)
Refine: GammaP	Refine: GammaP threshold. (0 - 30)
Refine: Tc or Rc pixel size	Use minimum pixel size of neighbour cameras (Tc) or current camera pixel size (Rc)
Verbose Level	verbosity level (fatal, error, warning, info, debug, trace)
Output	Output folder for generated depth maps

Table 2.7: Meshroom Depth MapNode

2.2.8 Depth Map Filter

This node is in charge of processing the DepthMap Node results s the original depth maps will not be entirely consistent. Certain depth maps will interprete to see areas that are occluded by other dept maps. Thus, this step of the pipeline isolates these areas and ensures depth consistency.

The components in the node are described on table 2.8

Name	Description
Input	SfMData file
Depth Map Folder	Input depth map folder
Number of Nearest Cameras	Number of nearest cameras used for filtering 10 (0 - 20)
Min Consistent Cameras	Min Number of Consistent Cameras 3 (0 - 10)
Min Consistent Cameras Bad Similarity	Min Number of Consistent Cameras for pixels with weak similarity value 4 (0 - 10)
Filtering Size in Pixels	Filtering size in Pixels (0 - 10)
Filtering Size in Pixels Bad Similarity	Filtering size in pixels (0 - 10)
Verbose Level	verbosity level (fatal, error, warning, info, debug, trace)
Output	Output folder for generated depth maps

Table 2.8: Meshroom Depth Map Filter Node

2.2.9 Meshing

The main purpose of this node is to construct a dense geometric surface representation of the scene. Initially, it fuses all the processed depth maps into a global octree and where it is compatible it will merge depth values into the octree cells. Afterwards, the node performs a 3D Delaunay tetrahedralization process. Subsequenly, a complex voting procedure computes the weighs on cells and weights the faces connecting the cells (Jancosek and Pajdla, 2011) as mentioned by Jancosek and Pajdla (2014).

A Graph Cut Max-Flow (Boykov and Kolmogorov, 2004) is applied to efficiencly reduce

the volume, which represents the extrated mesh surface. Furthermore, it filters bad cells on the surface and applies a Laplacian filtering process on the Mesh to remove local artifacts.

The components in the node are described on table 2.9

Name	Description
Input	SfMData file.
Depth Maps Folder	Input depth maps folder
Filtered Depth Maps Folder	Input filtered depth maps folder
Estimate Space From SfM	Estimate the 3d space from the SfM
Min Observations For SfM Space Estimation	Minimum number of observations for SfM space estimation. (0-100) 3
Min Observations Angle For SfM Space Estimation	Minimum angle between two observations for SfM space estimation. (0-120) 10
Max Input Points	Max input points loaded from depth map images (500**000** - 500000000)
Max Points	Max points at the end of the depth maps fusion (100**000** - 10000000)
Max Points Per Voxel	(500**000** – 30000000)
Min Step	The step used to load depth values from depth maps is computed from maxInputPts. Here we define the minimal value for this step, so on small datasets we will not spend too much time at the beginning loading all depth values (1- 20) 2
Partitioning	(singleBlock, auto)
Repartition	(multiResolution, regularGrid)
angleFactor	(0.0-200.0) 15.0
simFactor	(0.0-200.0) 1.0
pixSizeMarginInitCoef	(0.0-10.0) 2.0
pixSizeMarginFinalCoef	(0.0-10.0) 4.0
voteMarginFactor	(0.1-10.0) 4.0
contributeMarginFactor	(0.0-10.0) 2.0
simGaussianSizeInit	(0.0-50) 10.0
simGaussianSize	(0.0-50) 0.1
minAngleThreshold	(0.0-10.0) 0.01
Refine Fuse	Refine depth map fusion with the new pixels size defined by angle and similarity scores.
Add Landmarks To The Dense Point Cloud	Add SfM Landmarks to the dense point cloud.
Colorize Output	Whether to colorize output dense point cloud and mesh.
Save Raw Dense Point Cloud	Save dense point cloud before cut and filtering.
Verbose Level	verbosity level (fatal, error, warning, info, debug, trace).
Output Mesh	Output mesh (OBJ file format). mesh.obj
Output Dense Point Cloud	Output dense point cloud with visibilities (SfMData file format). densePointCloud.abc

Table 2.9: Meshroom Meshing Node

2.2.10 Mesh Filtering

This node filters and removes the unwanted elements from the resulting mesh. The components in the node are described on table 2.10

Name	Description
Input	Input Mesh (OBJ file format)
Filter Large Triangles Factor	Remove all large triangles. We consider a triangle as large if one edge is bigger than N times the average edge length. Put zero to disable it. 60 (1 - 100)
Keep Only the Largest Mesh	Keep only the largest connected triangles group (True/False)
Nb Iterations	5 (0 - 50)
Lambda	1 (0-10
Verbose Level	
Verbose Level	[‘fatal’, ‘error’, ‘warning’, ‘info’, ‘debug’, ‘trace’]
Output mesh	Output mesh (OBJ file format) internalFolder + ‘mesh.obj

Table 2.10: Meshroom Mesh FilteringNode

2.2.11 Texturing

The principle of this components is to create a texture of the generated mesh. If the mesh has no correspondent UV, it computes auomatic UV maps. It uses basic UV mapping approach to reduced the texture space as proposed by Levy et al. (2002).

For each triangle, it uses the visibility information associated to each vertex in order to, retrieve texture candidates. It filters the cameras that do not have a good angle to the surface to favourable fronto-parallel cameras to average the pixel values. Furthermore,

it uses a generalization of multiband blending as described by Burt and Adelson (1983). Thus, it averages more views in the low frequencies in comparison to the high frequencies.

The components in the node are described on table 2.11

MVS Configuration file	**.../mvs.ini**
Input Dense Reconstruction	Path to the dense reconstruction result (mesh with per vertex visibility)
Other Input Mesh	Optional input mesh to texture. By default, it will texture the result of the reconstruction.
Texture Side	Output texture size 1024, 2048, 4096, 8192, 16384
Texture Downscale	Texture downscale factor1, 2, 4, 8
Texture File Type	Texture File Type 'jpg', 'png', 'tiff', 'exr'
Unwrap Method	Method to unwrap input mesh if it does not have UV coordinates Basic (>600k faces) fast and simple. Can generate multiple atlases LSCM (<= 600k faces): optimize space. Generates one atlas ABF (<= 300k faces): optimize space and stretch. Generates one atlas
Fill Holes	Fill Texture holes with plausible values True/False
Padding	Texture edge padding size in pixel (0-100)
Max Nb of Images For Fusion	Max number of images to combine to create the final texture (0-10)
Best Score Threshold	0.0 to disable filtering based on threshold to relative best score (0.0-1.0)
Angle Hard Threshold	0.0 to disable angle hard threshold filtering (0.0, 180.0)
Force Visible By All Vertices	Triangle visibility is based on the union of vertices visiblity.True/False
Flip Normals	Option to flip face normals. It can be needed as it depends on the vertices order in triangles and the convention change from one software to another.
Visibility Remapping Method	Method to remap visibilities from the reconstruction to the input mesh (Pull, Push, PullPush).
Verbose Level	verbosity level (fatal, error, warning, info, debug, trace).
Output Folder	Folder for output mesh: OBJ, material and texture files.
Output Mesh	Folder for output mesh: OBJ, material and texture files. internalFolder + 'texturedMesh.obj
Output Material	Folder for output mesh: OBJ, material and texture files. internalFolder + 'texturedMesh.mtl
Output Textures	Folder for output mesh: OBJ, material and texture files. internalFolder + 'texture_*.png

Table 2.11: Meshroom Texturing Node

2.3 PCL

The **Point Cloud Library (PCL)** is a large scale, open project used for point cloud processing. The PCL framework contains multiple state-of-the art algorithms ranging from feature estimation,registration,filtering, model fitting to segmentation (Rusu and Cousins, 2011). PCL is a cross platform library and it is compatible with Windows, Linux and macOS.

The PCL framework is divided into a series of modular libraries such as:

- Filters
- Features
- Keypoints
- Kegistration
- kdtree
- octree
- Segmentation
- Sample Consensus
- Surface
- Recognition
- IO
- Visualization

These series of modular libraries have a series of embedded algorithms that can be used in scenarios such as: filtering outliers from noisy data, stitch multiple 3D pointclouds, segment relevants parts of the scene & point cloud, extract keypoints and compute image

descriptors to be able to recognize real world objects based on geometric appearence, among many other possible implementations.

The selected components and algorithms that were used in this project include:

- Statistical Outlier Removal Filter
- Pass Through Filter
- Voxel Grid Filtering
- Plane Model Segmentation
- KdTree
- Euclidean Cluster Extraction
- Principal Component Analysis
- Moving Least Squares
- Poisson Surface Reconstruction

2.3.1 Statistical Outlier Removal Filter

This component of the PCL library aims to remove noisy measurements (outliers) from a point cloud dataset using statistical analysis techniques (Rusu and Cousins, 2011).

Typically, Laser Scans produces point cloud datasets with various point densities.Similarly, measurement errors generate sparse outliers that can corrupt the results of the dataset. This, complicates process of estimating local point cloud characteristics in particular surface normals and curvature changes. Hence, leading to erroneous values, that can create failures in the point cloud registration process. Some of these irregularities in the dataset, can be addressed with a statistical analysis performed on each point's neighborhood, and removing the points that do not meet a specified criteria as mentioned by Rusu and Cousins (2011). The sparse outlier removal is built on the computation of the distribution of point to neightbors distances of the source dataset (Rusu and Cousins, 2011). For each point, it computes the mean distance from the points to all the corresponding neighbors. This process assumes that the resulting distribution is Gaussian with a standard deviation and a mean. Then all points whose mean distances are outside an interval defined by global distances (mean & standard deviation) can be expressed as an outlier and it can be removed from the dataset as suggested by Rusu and Cousins (2011).

Figure 2.3 illustrates the results of the sparce outlier analysis and removal process. The initial dataset is on the left and the processed result in on the right. Furthermore, the graph

on the right, illustrates the mean k-nearest neighbour distances in a point neighborhood, before and after the filtering process (Rusu and Cousins, 2011).

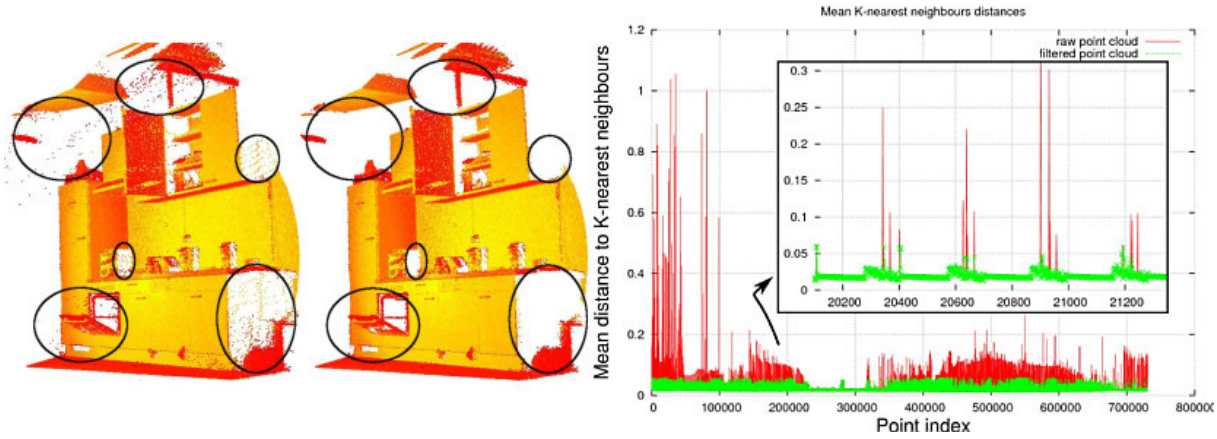


Figure 2.3: Removing outliers using a StatisticalOutlierRemoval filter (Rusu and Cousins, 2011)

2.3.2 Pass Through Filter

This component of the PCL library has embedded two main components of the framework. The **PassThrough** components uses the base **Filter** class methods to pass through all the data that satisfies certain contrains.

PassThrough passes points in a pointcloud based on constrains for one specific field of the point type (Rusu and Cousins, 2011). It iterates through the entire input pointcloud once, and automatically filters non-finite points, including the points outside the interval specfied by *setFilterLimits()*, that only applies uniquely to the configured field *setFilterFieldName()*. The component *setFilterLimits()* sets the numerical limits for the field for data filtering, whereas *setFilterFieldName()* configures the name of the field that will be used for data filtering.

Therefore, this component performs a filter along a given dimension, which essentially removes the values that are either outside or inside the configured range. The image 2.4 illustrates the process. The pointcloud has five points, after filtering the green points represents the remaining end result and the red points are the points that were removed by the filter.

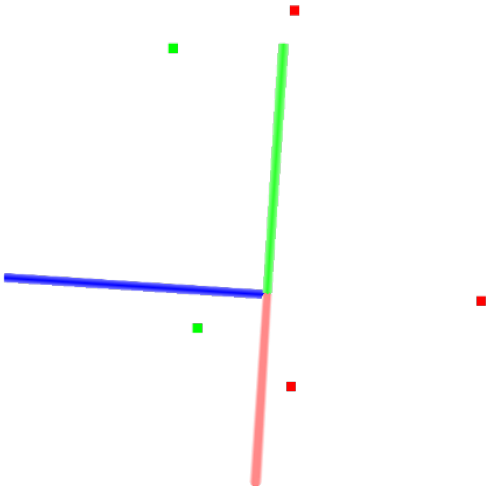


Figure 2.4: Filtering a PointCloud using a PassThrough filter
(Rusu and Cousins, 2011)

2.3.3 Voxel Grid Filtering

A voxel Grid illustraters a value on a regular grid in 3D space. A Voxel is an image of 3D space region which is limited by defined sizes, which has its corresponding nodal point coordinates in an accepted system, own form, owm state parameter that demostraters it belongs to some modelled object and it associated properties of the modelled region (Shchurova, 2015).

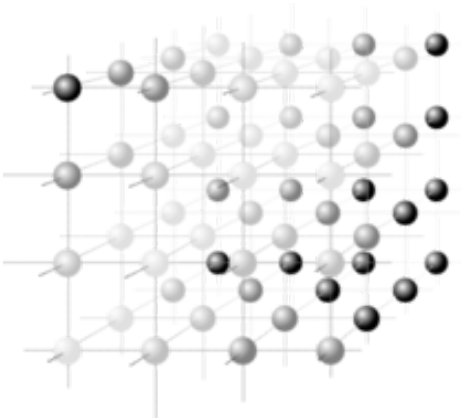


Figure 2.5: Illustration of a voxel grid containing color values
(Shchurova, 2015)

This component of the PCL framework aims to downsample (*Reduce the Number of Points*) of a point cloud dataset using a voxelized grid method. The **VoxelGrid** component, creates a 3D Voxel grid, which in common tearms refers to a set of small boxes in 3D space, over the input point cloud dataset. Once the its successfully converted into a voxel grid,

then in each voxel(*3D box*), all the existing points will be approximated (*Downsampled*) with their centroid as suggested by Rusu and Cousins (2011). This approach represents the underlaying surface of the point cloud dataset with high accuracy. The main component is to set a defined and Voxel leaf Size, which allows to set the Voxel Size and the Number of Voxel in the Voxel Grid. Therefore, directly influencing the downsampling process and the resulting processed point cloud.

2.3.4 Plane Model Segmentation

This component of the PCL librabry aims to perform a plane segmentation of a given set of points, which finds all the points within a point cloud data set that support a plane model. One of the components that this components uses is *pcl::ExtractIndices*, which aims to extract thhe indices from a point cloud. It is a filter that extracts a subset of points from a point cloud dataset, which are related to the indices output of a segmentation algorithm.

Similarly, it uses *pcl::SACSegmentation*, which represents the Nodelet segmentation class for Sample Consensus methods and model, in a way that create a Nodelet wrapper which can be used for generic-purpose SAC-based Segmentation (Rusu and Cousins, 2011). The *pcl::SACSegmentation*, creates an object and sets the model & method type. Furthermore, it specifies the "distance threshold", that is used to determine the distance for point in the model, in order to be considered an inlier. The used method in this project uses the **RANSAC** method. The main advantage of RANSAC is it's simplicity and robustness.

The estimated plane parameters are estimated with equation 2.2.1, where **a,b,c,d** are the model coefficient values.

$$ax + by + cz + d = 0 \tag{2.1}$$

Figure 2.6, illustrates the process below. The red points are the outliers, whereas the green points are the inliers of the found plane model.

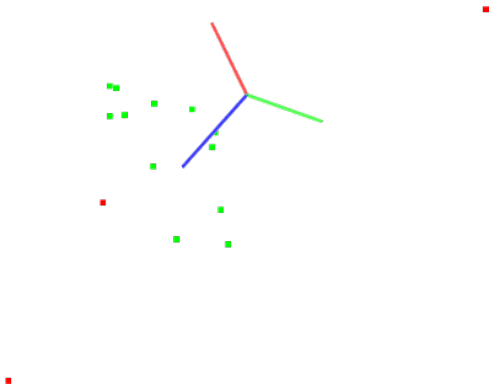
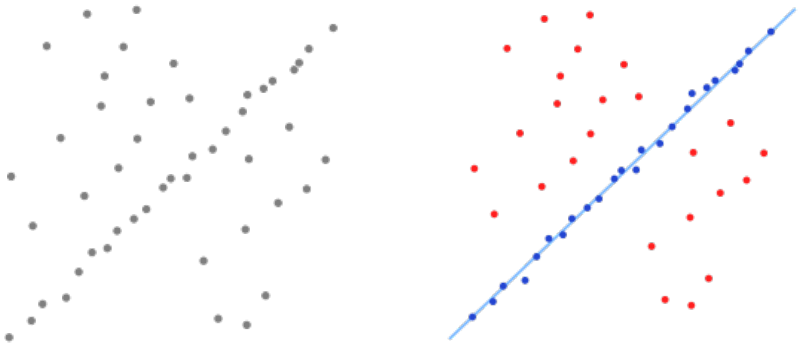


Figure 2.6: Illustration of Plane model segmentation (Shchurova, 2015)

RANSAC

Random sample consensus (RANSAC), is an iterative method that is used to estimate parameters of a mathematical model from a set of observed data that contains outliers (Strutz, 2016). Hence, it can be interpreted as an outlier detection method. It is an non-deterministic algorithm that generates an adequeate result with a specific probability,the probability increases as more iteration are performed.



(a) A data set with many outliers for which a line has to be fitted

(b) Fitted line with RANSAC; outliers have no influence on the result

Figure 2.7: Dataset Before & After RANSAC algorithm (Strutz, 2016)

2.3.5 KdTree

In this project Kdtrees are used to find the K nearest neighbour of certain points or locations and how to find these within an specified radius.

A K-D tree (*k-dimensional tree*), is a data structure that is commonly used to organize a certain number of points in a space with k dimensions as suggested by Rusu and Cousins

(2011). It is a binary search tree with other constrains imposed on it and are useful for searches for range and closest neighbours. As this project uses pointclouds, the Kdtrees would have three dimensions. Each level of a Kdtree splits all children along a specific dimension, utilising a hyperplane which is perpendicular to the corresponding axis. At the root of the KdTree, all children will be divided based on the first dimension, and each level down in the Kdtree divides on the next dimension, finally it returns to the first dimension once all other have been exhausted (Rusu and Cousins, 2011).

KdTrees are used extensively in many components of the project framework pipeline.

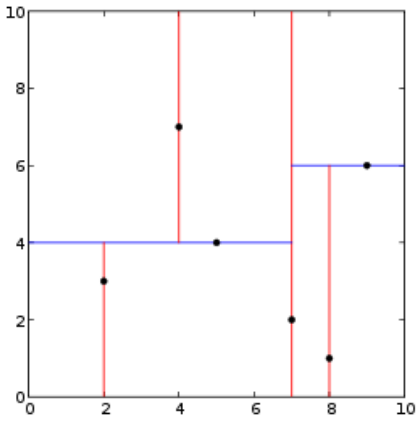


Figure 2.8: Example of a 2-dimensional k-d tree
(Rusu and Cousins, 2011)

2.3.6 Euclidean Cluster Extraction

The clustering method allows to divide a non-organized pointcloud dataset P into smaller components, in order to reduce the overall processing time for P. Point Cloud Data Clustering can be approached in an Euclidean manner and be implemented using 3D grid subdivisions of the space with fixed width boxes or a octree data structure. In this project, Kdtrees are used to find the nearest neighbours and implements a clustering technique which is similar to a flood fill algorithm (Rusu and Cousins, 2011).

The following Steps illustrates the clustering algorithm in this project with a Kdtree:

1. create a Kd-tree representation for the input point cloud dataset P;
2. set up an empty list of *clusters* C, and a queue of the points that need to be checked Q;
3. then for every point $p_i \in P$, perform the following steps:

- add p_i to the current queue Q;
- for every point $p_i \in Q$ perform:
  - Search for the set P_i^k of point neighbors of p_i in a sphere with radius $r < d_{th}$
  - for every neighbor $p_i^k \in P_i^k$, check if the point has already been processed, and if not add it to Q
- when the list of all points in Q has been processed, add Q to the list of clusters C, and reset Q to an empty list

4. the algorithm terminates when all points $p_i^k \in P$ have been processed and are now part of the list of point clusters C

Figure 2.9, illustrates a cluustering output of a dataset, where different components were divided into multiple clusters

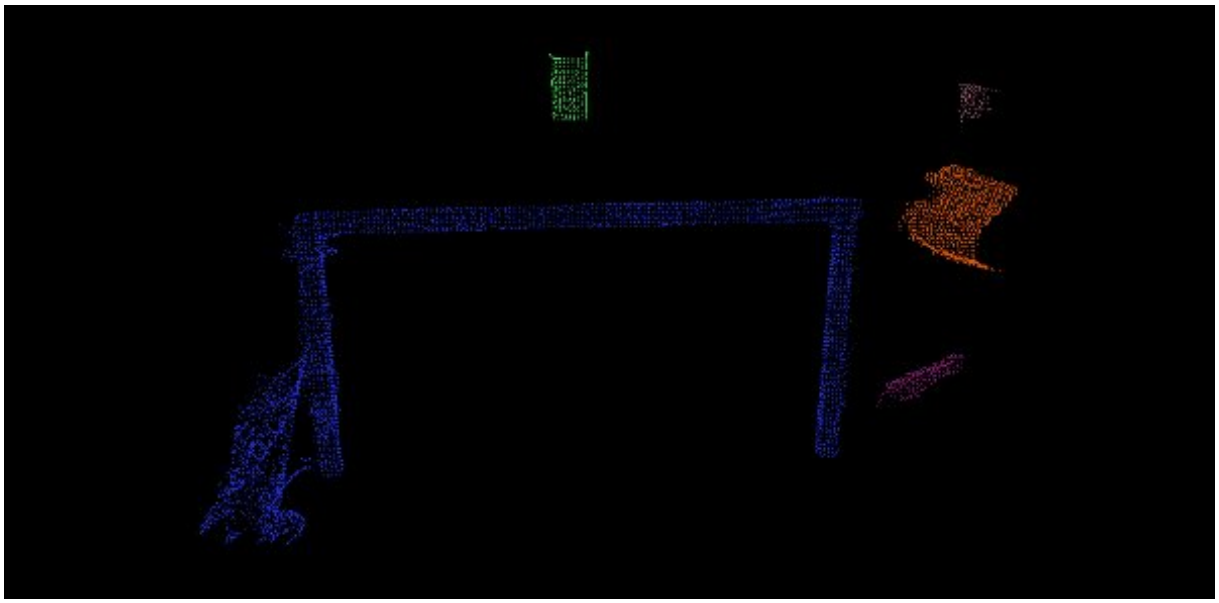


Figure 2.9: Example of point clouud cluster
(Rusu and Cousins, 2011)

2.3.7 Principal Component Analysis

The Principal Component Analysis (PCA) is the process of computing the principal components and use them to change the basis on data, using only the first few principal components and discarding the rest. The principal components is a squence of p unit vectors of a collection of points in a real coordinate space (Artac et al., 2002).

The principal components are extracted using a singular value decomposition method, which is applied on the covarient matrix of the centered input point cloud dataset. Once the PCA analysis is performed using the PCL library, it is possible to calculate the following components:

- Mean of input data
- Eigen Vectors: Ordered set of vectors that represents the final principal components and the eigen cartesian space.
- Eigen Values: These are the correspondent loading of the Eigen Vectors ina decensing order.

The Principal Component Analysis is used to determine the scale of the Two point clouds that are used in this project and properly adjust the final mesh.

2.3.8 Moving Least Squares

Moving Least Squares (MLS) is a method of reconstructing continous functions from a set of unorganized point samples through the computation of weighted least squares, biased towards the region around the point at which the reconstructed value is requested (Levin, 1998).

MLS can be modelled considering function f: $\mathbb{R}^n \rightarrow \mathbb{R}$ and a set of sample points $S = \{(x_i; f_i)|f(x_i) = f_i\}$. Then, it MLS approximation of degree m at the point x is $\tilde{p}(x)$ where $\tilde{p}$ reduces the weighted least square error $\sum_{i \in I}(p(x_i) - f_i)^2\theta(\|x - x_i\|)$ over all polynomials p of degree m in $\mathbb{R}^n$ as mentioned by Levin (1998).

The main objective of Moving Least Squares (MLS) is to smooth and resample noisy data in a surface reconstruction object. Some pointclouds have certain data irregularities, which can be caused by small errors in the distances measurements, and are complex to remove using conventional statistical analysis. It is important to account for complex surfaces and occlusions to create a complete model. Resampling algorithms such as MLS can be used to recreate the missing parts of the surface by a high order polynomial interpolation between the surroundings data points. Resampling, allows to correct the small errors and smooth the surface of a pointcloud dataset.

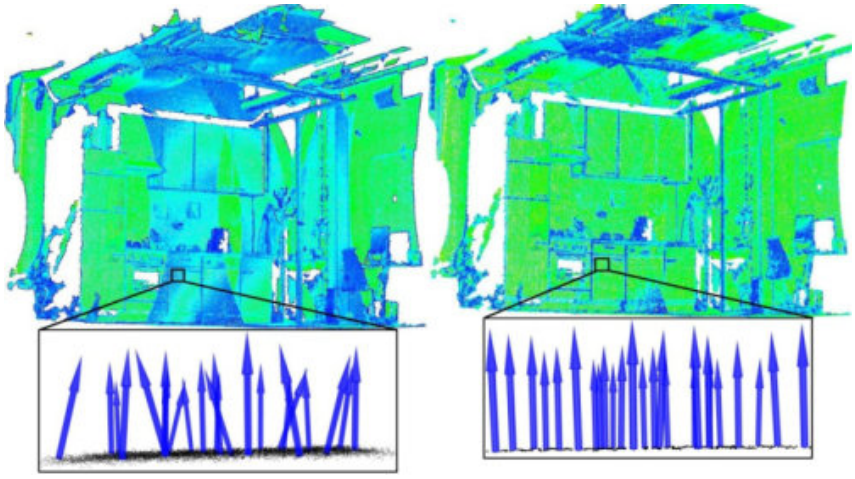


Figure 2.10: Moving Least Square Smoothing
(Rusu and Cousins, 2011)

On figure 2.10, on the left side, it is illustrated the effect of estimating the normals of a pointcloud, however, due to alighment errors the normals are noisy. On the right side of figure 2.10, the effects of the MLS algorithm is demostrated as it smoothes the surface of the pointcloud.

In order to approximate, the surface illustrated by the local neighbours of points $p1, p2, ...pk$ at a point q, it uses a bivariate polynomial function, which is defined on a robust modelled reference plane. Figure 2.11, shows the curvatures at each point with the eigen value relation before and after the resampling method.

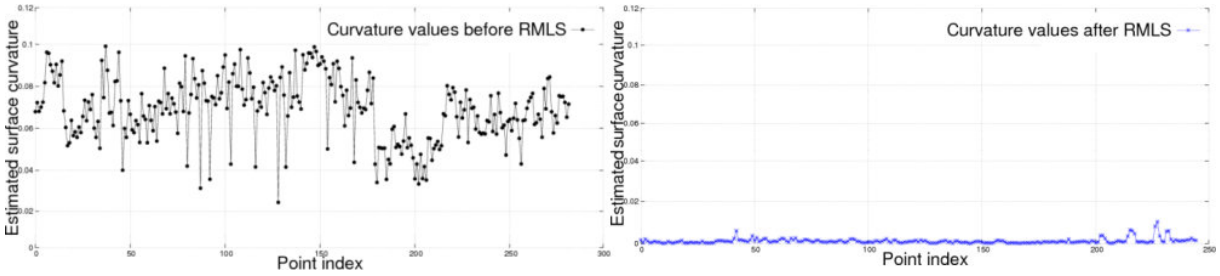


Figure 2.11: Curvatures of MLS before & after
(Rusu and Cousins, 2011)

Furthermore, the Moving Least Squares Component of the PCL library also allows for differnt upsample methods. These include

DISTINCT CLOUD Project the points of the distinct cloud to the MLS surface(Rusu and Cousins, 2011).

SAMPLE LOCAL PLANE The local plane of each input point will be sampled in a circular fashion using the $UpsamplingRadius$ and the $UpsamplingStep$ parameters(Rusu and Cousins, 2011).

RANDOM UNIFORM DENSITY The local plane of each input point will be sampled using an uniform random distribution such that the density of points is constant throughout the cloud - given by the $DesiredNumPointsinRadiusParameter$(Rusu and Cousins, 2011).

VOXEL GRID DILATION The input cloud will be inserted into a voxel grid with voxels of size $VoxelSize$.This voxel grid will be dilated ($DilationIterationNum$) times and the resulting points will be projected to the MLS surface of the closest point in the input cloud; the result is a point cloud with filled holes and a constant point density (Rusu and Cousins, 2011).

For this project, the **RANDOM UNIFORM DENSITY** method is used to upsample the MLS processed pointcloud. This method takes the parameters for a desired point cloud density within a fixed radius neighborhood as suggested by Ichim (2012). For each point, based on the density of the of the neighbors it will add more points on the local plane using a ramdom number generator. The random number generator will have a uniform distrubution and it will stop once the desired density is achived. Then it will replay the MLS algorithm in order to smooth the surface again.

Figure 2.12 illustrates on the left the implementation of MLS with **UNIFORM DENSITY** upsampling method on a pointcloud, whereas on the right if the raw pointcloud.

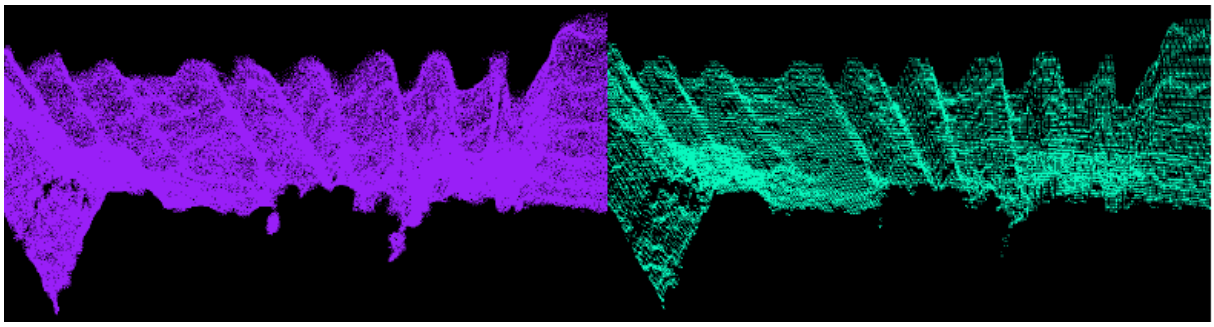


Figure 2.12: Uniform Density Upsample Method After & original
(Ichim, 2012)

2.3.9 Poisson Surface Reconstruction

In this project the Poisson Reconstruction algorithm is implemented , to recontruct the processed point cloud. The Poisson Reconstruction algorithm demostrates that the surface

reconstruction from a set of oriented points can be associates as a spatial Poisson problem. This formulation, utilizes all the points at once, with no need of using heuristic spatial partitioning or blending, hence, making it resilient to noise as mentioned by Kazhdan et al. (2006).

The main objective of this algorithms to reconstruct a smooth surface, which is based on a large number of points p_i from a pointcloud, where each point possesses an estimate of the local surface normal n_i It aims to create an implicit function f, whose value is zero at the points p_i and the gradient at points p_i is equal to the normal vector n_iKazhdan et al. (2006). The set of (p_i, n_i) is modelled as a continouos vector field V and the implicit function is found by integrating the vector field V. In complex calculations, it is possible to perform a *least-squares fit* to minimize the different between the gradient of f and VKazhdan et al. (2006).

Figure 2.13: Poisson Recontruction Example. On the left it it input pointcloud dataset, whereas on the right is output of the Poisson Surface Reconstruction Algorithm

(Kazhdan et al., 2006)

2.4 Open3D

Open3D is an open source library that supports fast development of software that handles 3D data. The Open3D frontend exposes outputs a set of carefully choosen data structures and algorithms in both Python and C++ (Zhou et al., 2018). The backend of the framework is highly optimized and it configured for parallelization. Open3D is compatible with Linux, macOS and Windows and it can be installed via source of via packages.

The core features include:

- Simple installation via conda and pip
- 3D data structures
- 3D data processing algorithms
- Scene reconstruction
- Surface alignment
- PBR rendering
- 3D visualization
- Python binding

In this project Open3D was used as an axiliary library used as contingency for Poisson Reconstruction. As in many circumstances the normals of a Pointcloud, might not be properly oriented *orient normals consistent tangent plane* allows to propagate the normal orientation with a minimum spanning tree as suggested by Zhou et al. (2018). After the normals is aligned it will excute the Poission Surface Recontruction algorithm (Kazhdan et al., 2006) and solves a regularized optimization problem to obtain a smooth surface mesh.

Figure 2.14: Open3D Poisson Recontruction Example
(Zhou et al., 2018)

Methodology

As mentioned in the initial section, this project is focused on engineering design. Hence, it will be focused on creating a prototype for a 3D scanner. This includes the hardware and the software for the 3D model reconstruction, from which this section will be focused on the 3D model data reconstruction of the project. The methodology will be divided into 3 sections that include:

- Data Acquisition
- Conceptual Design
- Design Implementation

3.1 Data Acquisition

As mentioned above the project can be described as an engineering design problem, whereby this approach includes the 3D software reconstruction model components for a 3D scanner. This includes gathering sensor data from the model that will be scanned. Many of the possible ways to gather data can be performed with commercially available sensors as proposed by Siena et al. (2018a).

Gautier et al. (2020) mentions that many 3D software reconstruction frameworks similar to Dynamic or SurfelWarp use commercially available sensors. One of the most common approach to get data for 3D scanning will be to use RGB-D cameras. The proposed sensor that will be used for data acquisition is the **"Intel RealSense D435i"**. This Sensor is an depth camera with a stereo solutoin which is used in a wide variety of applications which include:

- Robotics
- 3D Scanning

- Skeleton and Human Tracking
- Drones
- Objects Measurement
- Facial Auth

Therefore, making this camera ideal for this project application.

Figure 3.1: Intel RealSense D435i
(Intel, 2020a).

This model of camera has a proprietary software development library and SDK which is used to interact with the computer. This SDK will allow calibrating the camera and obtain certain calibration parameters that will be used for data acquisition. Furthermore, there is a software wrapper that will allow connecting with ROS (Robot Operating System). This ROS wrapper will allow interacting between the middleware, which is ROS, the camera, and the computer. With aid of this wrapper, it is possible to integrate all the components of ROS and the camera. One of the benefits of using ROS is that it allows recording a certain set of data based on the information that is being acquired by the sensor. The wrapper will start the camera based on different configuration and it will publish data both RGB and depth images in the following topics:

- /camera/color/camera_info
- /camera/color/image_raw
- /camera/depth/camera_info
- /camera/depth/image_rect_raw
- /camera/extrinsics/depth_to_color
- /camera/extrinsics/depth_to_infra1
- /camera/extrinsics/depth_to_infra2

- /camera/infra1/camera_info
- /camera/infra1/image_rect_raw
- /camera/infra2/camera_info
- /camera/infra2/image_rect_raw
- /camera/gyro/imu_info
- /camera/gyro/sample
- /camera/accel/imu_info
- /camera/accel/sample
- /diagnostics

The above topics will contain all the data that the sensors perceive in real-time. As can be seen from the above list, there will be data for the RGB, Depth sensors, Infrared sensors, IMU as well as all the extrinsic and status of the camera parameters. Furthermore, all this data will be processed with different formats such as raw or compressed. Based on the data perceived from these topics, it will be possible to create 3D point clouds in realtime and run 3D data reconstruction frameworks such as Surfelwarp as the sensors acquire and perceive the environment.

On the other side, ROS allows to record data from the above topics. From the above topics, it is possible to record the data messages as a "rosbag" or "bag", which is a file format in ROS for storing message data. Hence it will allow recording the data that the camera is sensing. Furthermore, these bags allow subscribing to specific topics, which implied that only the required topics will be stored for 3D data reconstruction and storing the data of the received message in an efficient file structure. The data acquisition process for the development phase will be performed by storing several rosbags, to be used in the development phase of the 3D reconstruction for the scanner.

3.2 Conceptual Design

The conceptual design will be focused on the development of the 3D software reconstruction for a 3D scanner. The Initial design will be to test with the field of view of the Intel RealSense D435i for both RGB image and Depth image sensors. From which for the Field of view (H $\times$ V $\times$ D) of the RGB corresponds to 69.4° $\times$ 42.5° $\times$ 77° ($\pm$3°), whereas the depth of the field of view for the Depth sensor corresponds to 86° $\times$ 57° ($\pm$3°)(Intel, 2020a).

Based on these results there will be a testing and evaluation phase, which involves evaluating what is the ideal configuration for the Camera. The ideal configuration will be based on which pose the sensor will be able to obtain the most accurate and useful data. Hence a mannequin will be used for this testing purpose from which the ideal pose for the sensor will be determined.

Once the ideal pose for the sensor is found, there will be the need to evaluate and test which is the fastest and most accurate way to obtain data of the model. All the testing purpose will be performed by recording several rosbags and visualise them. This has the intent to find the ideal location and number of all sensors that will be able to scan all the model. With this method, it will be possible to determine the speed and reliability of the Scanning process. Based results of this it will be possible to determine a prototype for the mechatronic design of the scanner as well as the methodology of how to fuse the data and run the 3D reconstruction framework.

The Design implementation of the 3D reconstruction software will be to adapt and adjust the 3D modelling data reconstruction frameworks. Hence, it includes developing the corresponding ROS nodes that will allow to use the data from multiple sensors and fuse it. With the fuse data, the intent is to run the 3D reconstruction framework which will output the Scanned model.

Once the process of generating the 3D scanned model is stable. The software will be ported to a small machine such as the Intel NUC, which is a small form factor computer. The software will be installed into the Intel NUC with the purpose to be incorporated with the mechatronic design of the 3D scanner. This will facilitate easy operation of the entire 3D scanning device.

Figure 3.2: Intel NUC
(Intel, 2020b).

Project Management

This section will provide guidelines and illustrate the management procedures that will be implemented into the feasibility and development of the software reconstruction for a 3D Scanner. It provides a projected timeline for deliveries and identifies specific methodologies implemented for the execution of the project from the beginning to the competition of all tasks. The project plan will be updated throughout the development of the project to reflect deliverables that may be affected by potential shortfalls. In order to prevent this, it will be necessary to circumvent any affected task and resolve it in an adequate timely manner. This will be a collaborative project, that involves creating the hardware device that will allow placing the sensors, the 3D data reconstruction of the Model, and the post utilization of the model for different uses such as testing virtual clothes. As the agreed allocated task was the 3D data reconstruction this section and document will be focused on the 3D software for data reconstruction.

4.1 Scope

There are multiple ways in which a 3D scanner is defined. Many connotations include creating the actual hardware of the device or how is the software going to be developed to create a 3D model. Based on this, the scope of this project component is to develop and deploy the software for 3D reconstruction of a scanning device as it was previously mentioned. The software will retrieve data from multiple sensors and cameras to then perform data fusion and create a 3D representation of the model that is scanned. Whereas the other components of the collaborative task will include the device creation as well as different applications for the obtained models.

The allocated 3D reconstruction part will include two phases to approach the engineering design problem that is targeted. The initial phase will be focused on gathering data from a wide variety of sensors that include RGB and Depth Cameras and test which is the

most effective way to collect data of the model. The second phase will be to develop a framework and methodology to fuse the acquired data from the sensors to create a 3D model representation of an object, which in this particular case will be a person. The project will span for over 35 weeks inclusive of Spring 2020, Summer Break 2021, and Autumn Session 2020 UTS academic semesters. This will include working with various teams that oversee the development of the hardware prototype as well as the uses of the model. The subsequent details are located in the sections below.

4.1.1 Project Specifications

It is important to outline all the specifications of the project in order to ensure, that the design will satisfy the purpose and it matches the scope of the project. As this is a collaborative project, the other team member will need to elaborate their corresponding details for their allocated sections. All of this will be reflected on the Work Breakdown Structure and Gantt Chart, however, the details for the 3D data model reconstruction will be explained below.

4.1.1.1 Sensing & Data Acquisition

There are multiple ways to achieve data acquisition that will satisfy the project. The proposed way would be to use sensors, especially, RGB and Depth Image Cameras The sensors should be able to provide with the following:

- RGB Colour images of the object or model
- Provide Depth images to get Distance for every point in the image
- Have a large enough field of view that could fit the entire object or model

The data acquired from these sensors should be reliable to ensure the fidelity of the final 3D reconstructed model. These data sensors will be converted to Point clouds via data fusion, where each point in the model will be mapped. Hence creating a 3D reconstruction model where its corresponding location in space is known. Therefore, the proposed sensor that will be used will be the Intel RealSense D435i. From which the SDK and ROS wrapper will allow to obtain the data and facilitate the process.

4.1.1.2 Data Fusion & Timing

Once the data is acquired from the sensors it will be required to fuse the data to be able to create the model. Data fusion is proposed to be performed in Realtime as the objects are scanned. This data fusion process will be performed by processing the data from multiple

sensors and running them to a 3D reconstruction framework such as SurfelWarp. Thus, creating a visualization preview of the object or person scanned.

On the other hand, the timing would be critical, as in most application the scanning time should not take more than a couple of minutes that could vary from 2 to 5 minutes. Therefore, ideally, the 3D reconstructed model from the scan should finalize within that time frame.

4.1.2 Project Overview & Deliverables

The project overview can be exemplified in figure 4.1 and it is associated with all the possible deliverables for the project

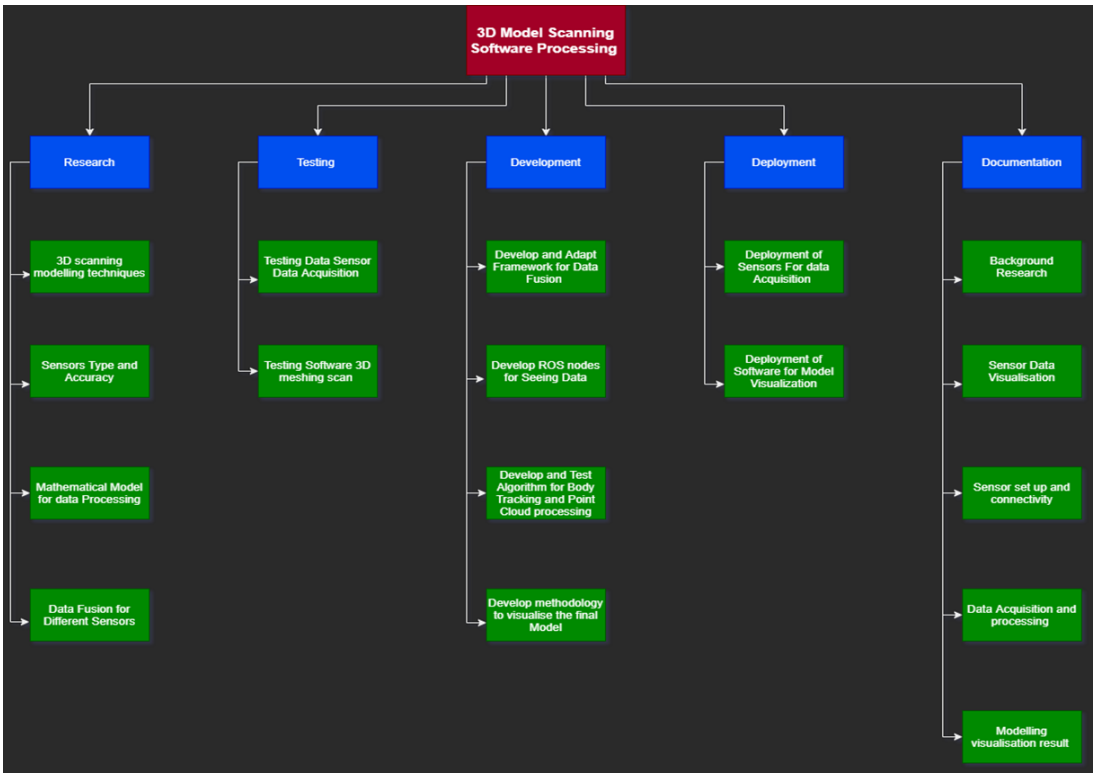


Figure 4.1: Constrained WBS for 3D data reconstruction component

Furthermore, it is imperative to demonstrate and illustrate the entire work break down the structure of the project. This will include other sections and components such as:

- Developing the hardware device to do the scanning methodology.
- Integrating the sensors for data acquisition and 3D reconstruction for the model.
- Uses the model for different applications.

The details will be located in the appendix on figure A.1

4.2 Project Timeline

The timeline shown in figure 4.2 depicts the problem-solving approach that is proposed in order to fully complete the optimum deliverable for the project. The below image summarises and synthesis the process in a waterfall approach. As there must be a certain number of deliverables that needs to be accounted for an agile approach would not satisfy the needs of the individual components. Once most components are developed with the waterfall approach, an agile methodology will be necessary to integrate all the other components of the project.

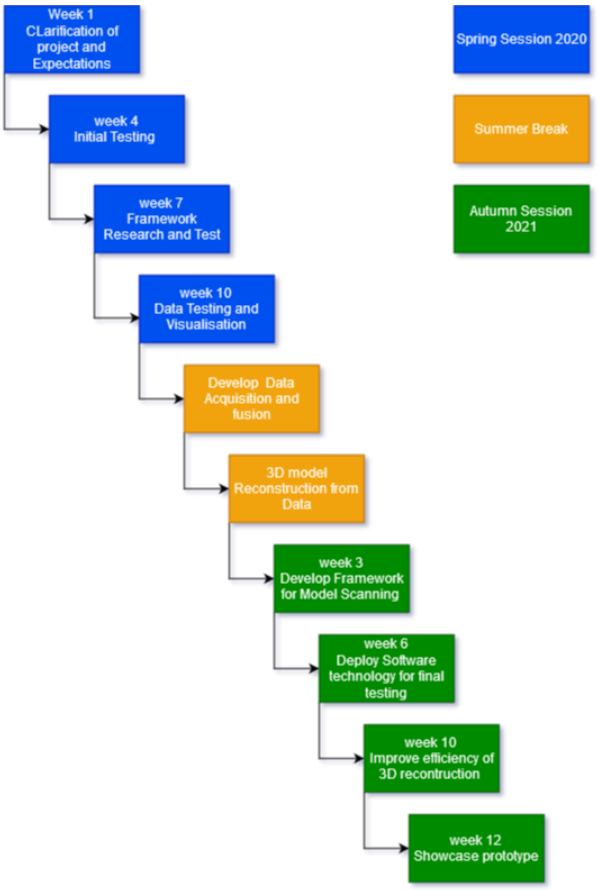


Figure 4.2: Transformation Between Depth Video to Warped canonical Model

These include:

- Hardware development of device
- 3D software for data reconstruction and modelling
- Uses and applications of 3D model

The proposed timeline from start to finish for the proposed project will be approximately 35 weeks. Including both UTS academic semester (Spring 2020 & Autumn 2021) and the corresponding university summer break holiday. All the detailed tasks and proposed deadlines are located in the Gantt Chart, located in the appendix. The proposed plan is inclusive of contingency scenarios as extra allocated days are added for different tasks. Any potential date changes will be updated accordingly in the plan, to keep the project completion as accurate as possible for the 3D reconstruction component.

4.3 Progress & Milestones

Any project must identify and define particular milestones for the project. Once the milestones are identified, they can be properly analysed to establish the corresponding requirements, processing, development and execution. After all these parameters have been established a corresponding timeline could be proposed. All these milestones will be considered for the 3D model reconstruction component.

This project component includes two particular and independent phases. Phase one will include the design proposal and data acquisition and processing, whereas phase two will focus on data fusion and 3D model data reconstruction. These phases could be subdivided five into smaller interdependent subprocesses that would contribute to the corresponding milestones that are part of the project deliverables. Thus, a holistic approach should be implemented with a waterfall implementation. The six major work areas include:

- Research and Conceptualization
- Testing
- Evaluation
- Development
- Deployment
- Documentation

The section below will delve into deeper detail of each project and illustrate the major milestones within itself. All do this will be illustrated for the 3D data model reconstruction component. All the other details for the other components are located in the work breakdown structure located in the appendix in figure A.1.

4.3.1 Milestones and Stages

4.3.1.1 Stage 1: Research and Conceptualization

The research component of this project will rely on exploring different robotics parameters and 3D data modelling reconstruction. The corresponding limitations will be analysed via research into different 3D reconstruction techniques that have been developed and what are the current developments in this 3D scanning field. The already existing solutions will aid to conceptualise the project to ensure optimal functionality that will produce an accurate 3D scanned model of persons or objects Potential solutions for software implementation

will be investigated in detail to grasp the basics of the different algorithm and how they behave in different scenarios.

Furthermore, different kind of sensors will be investigated to use the most appropriate equipment to grasp the data. Different data fusion frameworks will be analysed to make a collection of them in order to develop the data reconstruction prototype.

A recommendation will be made to the supervisor and team regarding optimal sensors that could be implemented and what type of data reconstruction could be utilised.

4.3.1.2 Stage 2: Testing

The testing phase for the 3D reconstruction component will be based on testing the different features and component that the d435i has available. The corresponding included SDK's will be used to see the camera parameters and different ways to grasp the Data. Furthermore, different tools will be tested to mesh raw data to test the reliability of the acquired data. On the other side, the meshing components will allow grasping the idea of how different techniques allow doing model meshing under different circumstances for the 3D data reconstruction framework.

All the parameters from the testing phase that include the test of single and multiple sensors for data acquisition. Similarly, an experimental test for sensors synchronization and calibration will be perform accordinly in the testing phase. Based on these parameters the evaluation of all the components will be perform accordingly before the development and deployment stages.

Moreover, there is the need to test how the different 3D reconstruct frameworks perform under different scenarions and conditions and evaluate the performance in order to start the development process.

4.3.1.3 Stage 3: Evaluation

The evaluation stage will be performed under the testing phase. From the obtained results there will be a corresponding team evaluation in which it will be assessed the use of single or multiple sensors. From this, it will be possible to obtain a clear insight into how many sensors will be necessary for the implementation. All these processes will be evaluated based on how the data is received and how it will be integrated with the final mechatronic prototype of the 3D scanner. This will be evaluated based on the requirements on how much data is necessary for the entire model as well as how much data can each sensor obtain based on the field of view.

Similarly, there will be a corresponding evaluation process on how to enable multiple

sensors synchronization. This will be tested and evaluated to ensure that the sensors will work simultaneously and that these will be able to acquire data synchronously. On the other side, it is crucial to test the accuracy of the sensors. Therefore it is a must to enable multiple sensor calibration. For this process, it will be necessary to obtain the camera calibration parameters. This includes obtaining the intrinsic and extrinsic parameters. Based on this it will be possible to obtain the distortion of the camera and it will be possible to calibrate each sensor accordingly. With these results, it will be possible to evaluate how multiple sensors will interact with each other and how the calibration process will be performed for the development and deployment stages

On the other side. it is crucial to assess the performance of Dynamic Fusion and SurfelWarp. These will include evaluating the performance under different scenarios and with different models. Also, the speed, accuracy and results will be assessed. Based on the results, it will be possible to choose the appropriate framework and to plan potential modifications that might need to be developed in the development stage.

4.3.1.4 Stage 4: Development

The development phase will consist of developing and adopting a framework for Data fusion. The data will be provided with the sensor that was chosen from the previous stage. The results obtained will allow to fuse all the data from all the components from the different sensors and visualise them into a single output.

Furthermore, there will be the development of the ROS nodes that will allow collecting the data in real-time that will be pass onto the data fusion framework. The ROS nodes will allow to properly map all the sensors and allow them to work simultaneously. Additionally, there will be the development and corresponding test for Body tracking of the scanned models and to convert the data from the sensors into Point clouds. These point clouds will be tested for accuracy of the Scanned data in the visualised model.

From this acquired and processed point clouds a methodology will be developed to work in conjunction with the Data fusion framework to visualise the final model. All these processes will be tested under different conditions to ensure that it will work under most circumstances and will allow a much easier deployment process.

4.3.1.5 Stage 5: Deployment

The deployment phase will be subdivided into two main tasks. The initial one consists of the deployment of all the sensors that will be utilised for data acquisition. This will include in deploying and installing the sensors into the actual hardware device that will

be used. These Sensors will be mounted and interconnected between then for enhanced sensing. This will be worked in conjunction with the hardware team.

Similarly there will be the deployment of the software tools that include the ROS nodes, Data fusion Framework and Model visualization methodology. All of this components will be deployed into the final device that is part of the hardware team.

4.3.1.6 Documentation

The documentation process is one of the crucial components of this project. Especially for the 3D data reconstruction component. All the corresponding stages of the research, testing, development, and deployment will have their proper documentation with a high level of detail. There will be the corresponding documentation regarding the background research of the project. It will include all the necessary research, use to determining the scope of the project and what are the current solutions for it.

Similarly, the documentation for all the sensor data visualization will include what type of sensor was used. It is characteristics, and what SDKs and libraries were used. The corresponding method of visualising the data will be documented as well as all the commands necessary for its use. Furthermore, all the necessary diagrams for connections and set up will be fully documented. As this will help massively the other teams.

There will be a detailed explanation of how the data acquisition process is carried out and what are the methodologies are used for data fusion. This also includes topics such as: what are the best scenarios and circumstances for using the developed technology. Finally, there will be the included detailed explanation of how the 3D model is generated. All other content such as risks a limitation will be included as the project is developed.

4.3.2 Gantt Chart

Please refer to Appendix, figure A.2, to visualise the Gantt Chart. The proposed Gantt Chart provides a detailed timeline, that includes dates and duration for each milestone. It will include milestones that are part of the hardware team and the use case team. Therefore, it will include all the proposed milestones for the entire project. It is important to know that the proposed timeline and dates might change, especially the ones that belong to the other teams (hardware and use case).

It is crucial to note the following:

- The work schedule and load differ between each academic semester. Thus, extra time was added to counteract this effect.

- The finalisation date and showcase were planned for the last week of the Autumn 2021 semester.
- There is a continuation of work through the UTS summer break with the purpose of maximising efficiency. This is reflected on the Gant Chart.

4.4 Resources

Resource planning is essential for any successful project. Different steps and different teams will require different resources and support. For this project, the principal resources can be split into human and technical.

4.4.1 Human Resources

As this project is non-confidential, it will not have any external relations. Hence, the main people involved in every component are the following:

Academic Supervisor – Dr Teresa Vidal Calleja

Teresa is the Deputy Head of School (Research), School of Mechanical and Mechatronic Engineering as well as Core Member for CAS - Centre for Autonomous Systems. Her Interest Include Robotic perception, automatic recognition, alternative sensing, visual SLAM, aerial and ground robot's cooperation, and autonomous navigation and manipulation.

Hardware Component lead – Asher Katz

Asher is the other capstone student of this project. He will oversee developing the hardware ad mechanism for the scanning device.

Use cases Component Lead – Mark Liu

Mark will oversee the use case of the project. He will use the 3D scanned models to try out clothing sizes virtually for different fashion items.

Cedric Le Gentil

Cedric will help in the process of creating the 3D scanner as well as facilitating the use case to adapt fashion items to the scanned models.

Nico Pietroni

Nico will be in charge of making the adaptation process between the 3D scanned models to the use case. Where he will develop the software for adding the clothes meshes to the 3D scanned model.

4.4.2 Technical Components

The technical components will include all the required sensors and software that will be implemented. The following resources are critical for the development, testing and deployment phases of the project.

4.4.2.1 Sensors

The proposed sensor that will be used for both testing and development is the Intel RealSense D435i. These components will need to be purchased to start with the project. Is estimated that the price for the D435i is 200 $ USD These sensors are crucial and will need to be imported from the US.

The corresponding software controllers and SDK for these sensors are available online and would only need to be installed and compiled. Depending on the outcome of the testing phase it will be necessary to determine the number of sensors that would be used. Hence the total overall price for the sensor is not fully established yet.

4.4.2.2 ROS

As mentioned previously ROS (Robotic Operating System) is a robotics middleware. It provides with the necessary services that are designed for a computer cluster that includes :

- Hardware abstraction
- Low-Level device control
- Synchronization for multiple sensors
- Real-time data acquisition of recording for sensors
- Sensor Data visualization in real-time.

ROS is free and can be installed easily in a Linux machine. All the proposed sensors have the corresponding ROS driver that will allow connecting the sensors with ROS with ease. This middleware will allow to collect, synchronise, and initialise all the sensors. As well as visualise all the acquired data in real-time. ROS allows an easy implementation of the framework for data fusion and Model reconstruction.

4.5 Uncertainties & Risk Control

All projects face some particular level of unforeseen risks and uncertainties. Good project planning will consider all possible areas of failure and introduce mitigation plans to try to control and prevent the consequences. All these uncertainties and risk matrix will be considered only for the 3D model data reconstruction component. The other components will need to be considered by the responsible respective teams.

The Uncertainties of the project float around the next parameters:

- How is the 3D data reconstruction going to work.
- How accurate is the data acquisition from the sensors.
- How long is the scanning process going to take.
- How long is the Testing and developing processes going to take.
- How is the deployment process going to work.
- How is the use case going to be applied for virtual clothing.

On the other side, the risk Matrix will explore all possible identified risks of the project. There is a severity scale from low to high that will illustrate the potential impact on the project and whether it will hinder the competition of itself. Additionally, the likelihood of each risk is analysed to reflect the priority risk mitigation. The risk mitigation table will be located in the appendix on table A.1.

4.6 Communication Management

The predicted communication plan was completed and developed after identifying all key stake holders, participants, communication channels for this endeavour. The major stake holders include:

- Capstone supervisor
- Hardware device capstone student
- User case team
- Engineering research preparation staff

As mentioned before the planned proposed time is expected to cover 35 weeks (27/7/2020 -30/6/2021). This includes both UTS academic semester as well as UTS summer break. The proposed communication plan will be located in the appendix on table A.2.

Progress Statement

The progress of this project is running accordingly to the plan of the Gantt Chart as reflected in figure A.2. This section will discuss in detail the findings of all the work that has been done to date. This includes researching the type of sensors that could be implemented such as the Intel RealSense D435i. Furthermore, there was a research stage that included to browse several 3D reconstruction frameworks and how these perform. On the other side, it was also necessary to research the components of the underlying frameworks and how it is constructed. Based on this, it was a must to investigate the dependencies are necessary to install these frameworks into the computer in order to start the 3D software reconstruction. Furthermore, all the project planning was developed. This includes creating the communication plan, risk matrix as well as the Gantt Chart and all the methodology that will be implemented on the project.

On the other side, several tests were performed with the purpose of gathering data of the model. These tests were performed with the purpose to test the different methods in which it was possible to gather data for a model. For these tests, a mannequin was used as the test model. Please refer to figure 5.1.

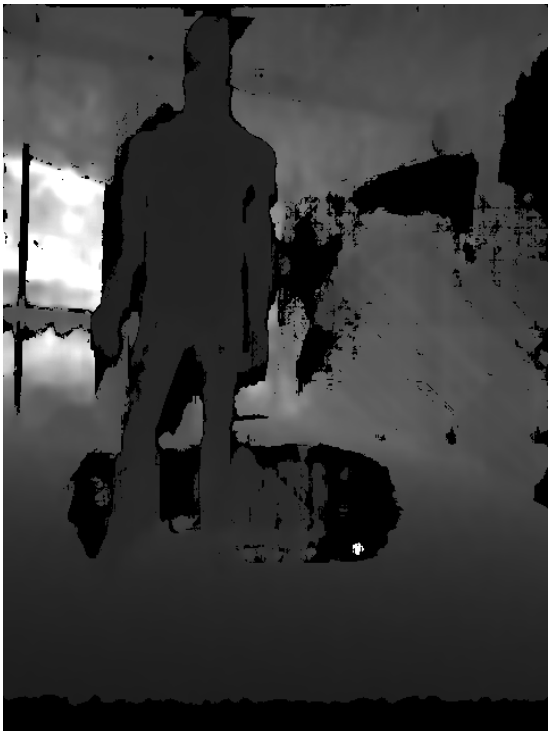
Figure 5.1: Manequin used for testing

From these tests, it was possible to obtain RGB and Depth images, that will allow seeing the Field of view of the sensor. Furthermore, it allowed to see and plan, the potential mechatronic design of the scanner and the number of sensors that could be used in the final design. The result of this test was that it was necessary to use a certain number of sensors in different positions in order to obtain as much information and data as possible for the scanned model.

In contrast, another test was performed with the purpose of gathering data from different positions. Hence, scanning a mannequin from different angles with the purpose of gathering the most data as possible. From these tests, it was possible to obtain the best configuration position for the camera, with the purpose of maximising the Field of View of the RGB and Depth Sensors. Furthermore, it allowed obtaining the 3D point clouds of the model in realtime, which allowed to have a preview of the scanned model.

Based on these tests it was also possible to determine the requires distance that the camera has to be from the model in order to obtain a preliminary radius of the scanner mechatronic design. This was done by observing the point clouds and via analysing the Sensor data in realtime. The RGB and Depth images acquire from these sensors can be observed on figures 5.2a and 5.2b respectively.

(a) RGB sensor data results



(b) Depth image sensor data results

Figure 5.2: RGB & Depth Image Data

Bibliography

Aber, J. S., Marzolff, I., Ries, J. B., and Aber, S. E. (2019). Chapter 3 - principles of photogrammetry. In Aber, J. S., Marzolff, I., Ries, J. B., and Aber, S. E., editors, *Small-Format Aerial Photography and UAS Imagery (Second Edition)*, pages 19–38. Academic Press, second edition edition.

AliceVision (2021). Meshroom manual.

Allene, C., Pons, J.-P., and Keriven, R. (2009). Keriven r.: Seamless image-based texture atlases using multi-band blending. pages 1 – 4.

Artac, M., Jogan, M., and Leonardis, A. (2002). Incremental pca for on-line visual learning and recognition. In *Object recognition supported by user interaction for service robots*, volume 3, pages 781–784 vol.3.

Baumberg, A. (2003). Blending images for texturing 3d models. *Proceedings of the British Machine Vision Conference*.

Boykov, Y. and Kolmogorov, V. (2004). An experimental comparison of in-cut/max-flow algorithms for energy minimization in vision. *IEEE transactions on pattern analysis and machine intelligence*, 26:1124–37.

Burt, P. and Adelson, E. (1983). A multiresolution spline with application to image mosaics. *ACM Trans. Graph.*, 2:217–236.

Cheng, J., Leng, C., Wu, J., Cui, H., and Lu, H. (2014). Fast and accurate image matching with cascade hashing for 3d reconstruction. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8.

Daanen, H. A. and Psikuta, A. (2018). 3d body scanning. *Automation in Garment Manufacturing*, pages 237–252.

Digumarti, S. T., Chaurasia, G., Taneja, A., Siegwart, R., Thomas, A., and Beardsley, P. (2016). Underwater 3d capture using a low-cost commercial depth camera. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9.

Fang, Z., Zhao, S., Wen, S., and Zhang, Y. (2018). A real-time 3d perception and reconstruction system based on a 2d laser scanner. *Journal of Sensors*, 2018:1–14.

Fernández Alcantarilla, P. (2013). Fast explicit diffusion for accelerated features in nonlinear scale spaces.

Fischler, M. and Bolles, R. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24:381–395.

Gao, W. and Tedrake, R. (2019). Surfelwarp: Efficient non-volumetric single view dynamic reconstruction. *CoRR*, abs/1904.13073.

Gautier, Q., Garrison, T., Rushton, F., Bouck, N., Lo, E., Tueller, P., Schurgers, C., and Kastner, R. (2020). Low-cost 3d scanning systems for cultural heritage documentation. *Journal of Cultural Heritage Management and Sustainable Development*, ahead-of-print.

Gonzalez-Rodriguez, A. (2020). Growing pains of ill-sized clothing for both consumers and brands.

Hirschmuller, H. (2005). Accurate and efficient stereo processing by semi-global matching and mutual information. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 807–814 vol. 2.

Ichim, A. I. (2012).

Intel (2020a). Depth camera d435i intel realsense depth and tracking cameras.

Intel (2020b). Intel nuc mini pcs2020.

Jancosek, M. and Pajdla, T. (2011). Multi-view reconstruction preserving weakly-supported surfaces. In *CVPR 2011*, pages 3121–3128.

Jancosek, M. and Pajdla, T. (2014). Exploiting visibility information in surface reconstruction to preserve weakly supported surfaces. *International Scholarly Research Notices*, 2014:1–20.

Jo, W., Kannan, S. S., Cha, G.-E., Lee, A., and Min, B.-C. (2020). Rosbag-based multimodal affective dataset for emotional and cognitive states.

Kazhdan, M., Bolitho, M., and Hoppe, H. (2006). Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, page 61–70, Goslar, DEU. Eurographics Association.

Keller, M., Lefloch, D., Lambers, M., Izadi, S., Weyrich, T., and Kolb, A. (2013). Real-time 3d reconstruction in dynamic scenes using point-based fusion. page 8.

Kneip, L., Scaramuzza, D., and Siegwart, R. (2011). A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. pages 2969–2976.

Lepetit, V., Moreno-Noguer, F., and Fua, P. (2008). Epnp: An accurate o(n) solution to the pnp problem. *International Journal of Computer Vision*, 81:155–166.

Levin, D. (1998). The approximation power of moving least-squares. *Math. Comput.*, 67(224):1517–1531.

Levy, B., Petitjean, S., Ray, N., and Maillot, J. (2002). Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph.*, 21:362–371.

Li, Y., Wang, S., Tian, Q., and Ding, X. (2015). A survey of recent advances in visual feature detection. *Neurocomputing*, 149:736–751.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–.

Magnenat-Thalmann, N., Seo, H., and Cordier, F. (2004). Automatic modeling of virtual humans and body clothing. *Journal of Computer Science and Technology*, 19(5):575–584.

Moulon, P., Monasse, P., and Marlet, R. (2012). Adaptive structure from motion with a contrario model estimation.

Muja, M. and Lowe, D. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. volume 1, pages 331–340.

Murcia, H., Monroy, M. F., and Mora, L. F. (2018). *3D Scene Reconstruction Based on a 2D Moving LiDAR.*

Newcombe, R. A., Fox, D., and Seitz, S. M. (2015). Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Nister, D. (2004). An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):756–770.

Nistér, D. and Stewénius, H. (2006). Scalable recognition with a vocabulary tree. volume 2, pages 2161 – 2168.

Otero, I. (2015). *Anatomy of the SIFT method*. PhD thesis.

Rusu, R. B. and Cousins, S. (2011). 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China.

Scharstein, D., Szeliski, R., and Zabih, R. (2001). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In *Proceedings IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001)*, pages 131–140.

Shah, R., Deshpande, A., and Narayanan, P. (2014). Multistage sfm: Revisiting incremental structure from motion. pages 417–424.

Shchurova, C. I. (2015). A methodology to design a 3d graphic editor for micro-modeling of fiber-reinforced composite parts. *Advances in Engineering Software*, 90:76–82.

Siena, F., Byrom, B., Watts, P., and Breedon, P. (2018a). Utilising the intel realsense camera for measuring health outcomes in clinical research. *Journal of Medical Systems*, 42.

Siena, F., Byrom, B., Watts, P., and Breedon, P. (2018b). Utilising the intel realsense camera for measuring health outcomes in clinical research. *Journal of Medical Systems*, 42.

Slavcheva, M., Baust, M., and Ilic, S. (2018). Sobolevfusion: 3d reconstruction of scenes undergoing free non-rigid motion. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Spahiu, T., Shehi, E., and Piperi, E. (2014). *Extracting body dimensions from 3D body scanning.*

Strecha, C., Fransens, R., and Van Gool, L. (2006). Combined depth and outlier estimation in multi-view stereo. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2394–2401.

Strutz, T. (2016). *Data Fitting and Uncertainty (2nd edition).*

Sturm, J., Bylow, E., Kahl, F., and Cremers, D. (2013). Copyme3d: Scanning and printing persons in 3d. *Lecture Notes in Computer Science*, pages 405–414.

Treleaven, P. and Wells, J. (2007). 3d body scanning and healthcare applications. *Computer*, 40(7):28–34.

Vedaldi, A. and Fulkerson, B. (2010). Vlfeat: An open and portable library of computer vision algorithms. volume 3, pages 1469–1472.

Yu, G. and Morel, J.-M. (2011). Asift: An algorithm for fully affine invariant comparison. *Image Processing On Line*, 1.

Zhou, Q.-Y., Park, J., and Koltun, V. (2018). Open3D: A modern library for 3D data processing. *arXiv:1801.09847*.

Appendix

Risk	Affected Phase	Severity	Likelihood	Consequences	Mitigation Plan
Lack of resources access for sensors	Research, Testing, Development and Deployment	Medium-High	Unlikely	Time setback and delays in the milestones for the delivery which could hinder the final demonstration and showcase.	Ensure that all sensors and hardware are purchased accordingly . Ensure that all SDK and driver are installed in workstations
Pandemic Progression	All Phases	Medium	Very Unlikely	The situation might trigger further adjustments to the scope of the project. As well as supervision availability	Moving all communications channel to MS teams and Zoom.
Supervisor unavailability and Low Guidance	All Phases	High	Very Likely	Lack of support could trigger a delay in the project competition	Establish as Well defined communication method and Look for other sources of guidance.
Excessive testing for Data Fusion framework and data modelling	Development and deployment	Medium	Unlikely	Loss of time for the final integration process for the 3D data reconstruction model.	Establish a reasonable time frame for testing all data visualization techniques.
No Finalization of developing framework for Data fusion and 3D reconstruction	Development	High	Unlikely	This will cause that the project wont finalise successful at the end	Try to gather as much sources as possible in order to get finalise the project.
Inability to fully integrate with the components from the other teams	Deployment	Medium	Unlikely	The consequence is that the project will get delay and the components will work independently but not combined.	Try to develop a an organised schedule that will give the proper timing for the correct integration with the other components and teams.

Table A.1: Risk Matrix

Subject	Contact	Channel	Discussion Topic	Frequency	Notes
Supervisor Meeting 1. Initial Project Discussion	Dr Teresa Vidal Calleja	Zoom	Introduce the proposed topic and discussion about how the components were labelled and divided.	One time occurrence 15/7/2020	Agreement on division of different components of the project. Establish requires skills.
Initial team discussion	Teresa Vidal Calleja, Asher Katz, Mark Liu	Zoom, Teams	Initial discussion about different components and Scope of the project	One time meeting 29/7/2020	Introduction of all team members and project discussion.
Team Meeting	Teresa Vidal Calleja, Asher Katz, Mark Liu	Zoom Teams	Weekly Discussion for all topics regarding the project and advancement.	Weekly 1 hour meeting every Tuesday until 20/6/2021	General Discussion about project status and updates.
Initial Data Acquisition Testing	Teresa Vidal Calleja, Asher Katz, Mark Liu	In Person	Initial scanning of model. Initial glance at sensor model.	One time occurrence 25/8/2020 3 Hours meeting	Initial testing with Intel real sense d435i
Engineering Research Preparation consultation	Xi Jin	Zoom	Assessment Task consultations and feedback	28/7/2020 18/8/2020 29/9/2020	Assessment task related questions and overview
Discussion on Software and hardware for Scanner	Asher Katz	TBA	General discussion and planning for integration and deployment	TBA	Lay out and planning or integration of the hardware component and 3D data reconstruction component
Final Report Submission	Teresa Vidal Calleja	Face to Face or zoom	Get final approval of project from supervisor	Single occurrence TBA	Get Final mark and sign from supervisor.
Project Showcase	Academic panel	TBA	Presentation of project	TBA	TBD
TOTAL HOURS COMPLETED				Expected	Actual
				65 Hours	TBD

Table A.2: Comunication Plan

Figure A.1: WorkBreakDown Structure

Figure A.2: Gantt Chart