3D Reconstruction application in Quarrying industry

Ágata Filipa Lopes de Barros

agata.barros@tecnico.ulisboa.pt

Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal

July 2021

Abstract

For the past 20 decades, the desire to produce 3D models of the world from 2D images has been researched in order to recover 3D point clouds based on Structure from motion and Multiview stereo methods. Meanwhile, the exploitation of minerals through quarrying, has been a lucrative market and a major contribution to the infrastructures of the cities since before the 18th century. The aim of this study is to apply 3D reconstruction approaches to limestone quarries. Building 3D model of quarry limestones from 2D images to be implemented in the quarry industry in favor of improving and modernizing the cleaving process through computer vision. The 2D images were acquired by a stereo camera pair, in a similar uncontrolled environment of an open-mine pit. In order to create fitting 2D images for structure from motion applications, it was necessary to explore different preprocessing approaches. Preprocessing methods were analyzed, and a gamma correction with a histogram equalization was deemed to be the most successful approach. Once the images were adjusted in terms of brightness and contrast, different SFM and MVS pipelines were tested to conclude which is the best available software application and its correspondent algorithm sequence. Although, the open-source software, Meshroom, was determined to be the best 3D reconstruction system, Agisoft Metashape can be more appropriate for industry applications due to its community support and convenient renewability. Nevertheless, Meshroom allows to study different matching algorithms, with known and unknown intrinsic and extrinsic parameters to rule the most effective approach.

Keywords: 3D reconstruction, Computer vision, Quarry industry, Structure from motion (SFM), Multi-view Stereo (MVS)

Introduction

Computer Vision is a scientific discipline in constant improvement, described as the acquisition, processing and analyzing digital images or videos to filter crucial information through mathematical models [1]. University studies have been focused in simulating a human visual system, since late 1960 [2], and the following decades proved to be decisive in the development and applications of this field by deriving a three-dimensional structure from images and therefore, creating the foundation for several computer vision algorithms, such as, edge detection, optical flow and motion estimation. From there on, researchers were able to establish new and more complex mathematical concepts, including the scale-space theory, in order to forge sparse 3D reconstructions of scenes. This led to a better understanding of camera calibration and optimization methods already established, like bundle adjustment. During the 1990s, techniques such as multi-view stereo solved the correspondence problem to advance from sparse to dense 3D reconstruction from multiple images [1].

From an engineering perspective, Computer vision is another study to create and implement automated technologies to improve efficiency and reliability of well-established processes. Computer vision aims to build autonomous systems to substitute human tasks which operate from human vision. On the other hand, quarrying industry is the extraction of natural stone, gravel and sand business. The process of extraction aggregates starts by cutting and removing big blocks of stone from open-pit mines, afterwards, the stone is transported to another machine to be cut in order to end up with a marketable stone block. The second phase can be described as a cutting machine operated by a worker, whose purpose is to initialize the engine for the cutting structure to travel through the different stone blocks in a row, for approximately 30 minutes, but also to identify the exact location where the cut should be placed and placing the cutting apparatus on this location to be able to perform the groove. The cutting location is analyzed by the worker, through experience knowledge, to remove flaws and to obtain a geometrically pleasing stone block for commercial applications.

The idea would be to create an autonomous system that would capture a video of the limestone blocks in line on the machine to create a 3D reconstruction model of each limestone block and thus, build a more reliable and efficient process of analyzing block stones. Therefore, this thesis statement is to implement and compare methods for full 3D reconstruction on a Limestone block from images, captured in an uncontrolled environment, addressing the requirements, effectiveness and reliability, for each step of the 3D reconstruction. This assignment aims to answer a main question:

How viable is to reconstruct a 3D model of a limestone taken in an uncontrolled environment for quarrying industry applications?

In order to answer this question, it is imperative to answer parallel questions, such as: What is the best software to perform 3D reconstruction of limestones? What is the best method to preprocess the images for better 3D model quality but also to allow the pipelines to reconstruct the scene? What is the best combination of algorithms within the chosen software? What is the duration of a 3D reconstruction of a limestone line? Is it possible to improve the duration and the quality of the 3D model by applying known intrinsic and extrinsic parameters?

Literary Review

An image-based 3D reconstruction is a robust, inexpensive and flexibly automated approach to acquire data in the form of 2D images to reach into a virtual 3D model [3][4]. A reconstruction made using multiple cameras can achieve better geometric data, a surface texture of visually realistic models, increase robustness and decrease image noise compared to a two-view camera reconstruction approach [5]. The approach of reconstructing a 3D model of an object or scene from correspondences between sequences of images taken from multiple viewpoints is the Multi-view geometry method. Multi-view geometry establishes correspondences between points in different images from the 3D position in a scene of the image plane [1]. A SFM approach estimates information for a multi-view stereo method to be applied in case the camera parameters and the camera location and orientation are unknown.

The process of 3D model building has several steps: From a camera alignment procedure, in which a feature detection and description algorithm, a feature matching algorithm and pose estimation methods. When the camera poses are known and detected features are matched, the Direct Linear Transform (DLT) is used for triangulation. Once a sparse point cloud is generated, it is possible to create a dense point cloud through MVS approaches. A 3D surface is built from surface reconstruction methods, such as, the Delaunay triangulation and the Poisson surface reconstruction algorithm. Although, these steps complete the main implementation of 3D reconstruction, the quality of the images is the first obstacle in3D reconstruction. If the images are captured in an uncontrolled environment, the quality of the images might need to be improved, through contrast and brightness enhancement, before reconstructing a 3D model of the scene. There are a variety of methods to enhance contrast and brightness that can be implemented in *Python* such as the histogram equalization, CLAHE and gamma correction [6]. According to [7], a good contrast enhancement can also be achieved by combining gamma correction to avoid excess brightness and a histogram equalization method, such as , CLAHE, to provide a good contrast to the images

Camera alignment

Camera alignment consists in 5 stages: Feature detection, feature matching and camera pose estimation. Feature detection is usually the first step in image processing with the purpose of identifying relevant pieces of information in the image. Feature matching matches the features detected and camera pose estimation determines the position of the cameras.

Feature detection and extraction

The pipelines adopted on this assignment implement SIFT or its variations to accomplish feature extraction. SIFT is patented by the university of British Columbia and described by one of the most influential research in Computer Vision [8]. SIFT stands for scale invariant feature transform which can be applied to different size images, different depth and the most advantageous characteristic of this feature detection algorithm, it is scale invariant. SIFT transforms image data into scale-invariant coordinates. The goal of this algorithm is to extract invariant features to be able to correctly match against a large database of features from many images. The features are invariant to image scale, rotation and robust to change in 3D viewpoint, noise and affine distortion [8]. As it was mentioned previously, SIFT is divided into 4 main steps: Scale space extrema estimation, keypoint localization, keypoint orientation assignment and the generation of a local image descriptor for each keypoint [8].

Feature matching

According to Lowe in [8], the features between two images are matched by identifying its nearest neighbor in the database of keypoints. The definition of nearest neighbor is the keypoint with the minimum Euclidean distance. Some features won't have any correct match for not being detected in the previous step or due to noise, therefore, the most effective measure to match these features is to compare the distance of the closest neighbor to the second closest neighbor. The second closest match, sometimes, may be near to the first due to noise, therefore the ratio between the closest distance and the second closest distance is analyzed. If the ratio is greater than 0,8 these matches are rejected, eliminating 90% of false matches and discarding only 5% of correct matches [8]. This approach is computationally expensive, especially if the input is a large number of images, in this case this brute-force method can be replaced by a Cascade hashing method or an Approximate Nearest Neighbor (ANN). The approximate nearest neighbor method preprocesses the points into a data structure to assign the nearest points of all the points to a query point q and the distance between two points can be define in Euclidean distance. The cascade hashing [9] method uses hashing to convert all feature into binary code to conduct a fast speed operation.

The image matching strategies [10] compared in this study are the following: The exhaustive matching, the sequential and the vocabulary tree matching. The exhaustive matching compares each image with all images, matching all image pairs exhaustively. The sequential image matching depends on the location relationship between cameras in the world coordinates to efficiently match image in sequential order. Vocabulary tree matching matches every image according to its visual nearest neighbors, therefore, this method indexes the images ranked by similarity of its visual vocabulary.

RANSAC [11] is an abbreviation for Random Sample Consensus. RANSAC is an iterative method that detects outliers and estimates a mathematical model with no outlier influence. This method is used to detect and discard outliers of matched features. RANSAC can remove false matches based on the epipolar geometry of the image pair. The majority of good samples allow a single model, however bad samples don't consistently agree with a single model, therefore, instead of detecting the bad inconsistent samples, the objective is to group samples that look the same and fit a model only to group the inliers. This method randomly selects the minimal points to sample in order to provide a concept of inliers. RANSAC algorithm adopts the following structure: Random sampling, model building, thresholding and inlier counting

Pose Location Estimation

This is the last step to build a sparse point cloud and consists in determining the camera parameters of each image, including intrinsic parameters and extrinsic parameters based on all the information gathered until this stage. Besides, computing the camera parameters, this step also determines the 3D locations of the point by using the feature points matched and the camera parameters through a triangulation algorithm.

Camera motion estimation

Epipolar geometry [1] is the projective geometry between two views. Assuming a static scene with a minimum of 2 views and a set of point correspondences between both views with known intrinsic parameters of the camera of both views. Epipolar geometry aims to find the 3D points based on the parameters. It determines the extrinsic parameters, algebraically, and the 3D coordinates. Considering a point in 3D world as X, where its projection onto the first view results in a 2D coordinate x_1 and its projection onto the second view results in a 3D coordinate x_2. Epipolar geometry can describe the correspondences between the matched feature point of the image pair which are: The essential matrix and the fundamental matrix. There are various methods to compute the camera parameters. The Five-point [12] algorithm is used to calculate the extrinsic parameters while the eight-point algorithm [13] estimates the intrinsic and extrinsic parameters. Usually, the Five-point algorithm is computed when the intrinsic parameter is one of the inputs on the 3D reconstruction process. This problem can also be called a Perspective n-point problem which consists in analyzing the correspondences between the 3D coordinates of an object onto the 2D coordinates of an image, in order to estimate the pose of the camera in relation to the set of n 3D points. Normalized 8-point algorithm [13] is a method that solves the fundamental matrix, assuming it can only be solved if there are 8 point correspondences to compute the fundamental matrix. The most common methods that solve PnP problems are the P3P method and the EPnP method. Both methods are frequently developed in open-source software.

Triangulation

Once the camera parameters are known it is possible to compute the 3D locations of the matched points by utilizing the feature points with respective correspondences and the camera parameters and positions of each image. The DLT [1] is an algorithm that formulates a homogeneous linear system and solves a set of variables, carrying similarity relations to solve triangulation problems.

Model parameters correction

Before generating the sparse cloud of the scene, most 3D reconstruction pipelines correct the camera parameters of each image and the 3D points obtained in the previous step through the Bundle adjustment method or its variation. Bundle Block adjustment or Bundle adjustment was originated in the photogrammetry field with the objective of minimizing the reprojection error between the location of the image measured points and the predicted image points. This approach focuses in optimizing the camera pose, the intrinsic calibration and the location of the 3D point to obtain an optimal 3D reconstruction. This optimization problem is solved based on a non-linear least-squares algorithm. Multicore Bundle Adjustment [14] consists in applying CPU and GPU parallelism to achieve a faster bundle adjustment method and solve larger problems A very established library is Ceres Solver. Ceres Solver is an open source C^{++} library for modeling and solving large optimization problems such as a nonlinear least squares problem.

Once the model parameters are corrected, the sparse point cloud can be built, and the 3D reconstruction process can proceed to a Dense reconstruction process.

Dense Point Cloud Reconstruction

Dense point cloud is the representation of depth maps. Depth maps are images represented in gray pixels of intensity between 0 to 255 value, in which, 0 corresponds to the black pixels and the farthest away from the source that took the images and 255 corresponds to the white pixels that are near the sensor. The Dense Cloud reconstruction is implemented though Multi-view stereo approaches such as CMVS[15]/PMVS[16] and SGM [15]. CMVS/PMVS can be divided into two steps: The Clustering Views for Multi-View Stereo (CMVS) and the Patch-based Multi-View Stereo (PMVS). The CMVS is adopted when the 3D points and the camera pose are known to group similar image with the intention of optimizing the MVS process, afterwards the PMVS can be computed to generate the 3D model.

Surface Reconstruction

Surface reconstruction states a problem of conversion, initiating with a dense point cloud set as an input to a surface. This process recovers the topology and geometry of the object or scene that is being reconstructed. There are two main methods applied to reconstruct the surface of the limestone block: The Poisson Surface Reconstruction [17] and the Delaunay Triangulation [18].

Textured Reconstruction

Textured reconstruction [19], as the last step of 3D reconstruction, offers color information mapped onto the geometry. This step generates texture parametrization to input in the 3D surface model. The color information is obtained from the calibrated cameras to generate a new

texture mapping through detailed color encoding “per-vertex”. The texture mapping consists in a parametrization that associates a pixel in a 2D image to each point on the 3D surface and the color is stored in an image as texture map.

Problem formulation

The 3D reconstruction of the limestone using a passive method targets a Structure from motion and multiple view stereo algorithms for both open source and commercial pipelines. After acquiring the images, the software starts by applying a structure from motion approach. This phase generates a sparse point cloud. Once the camera locations are known the next step uses the intrinsic and extrinsic parameters of the camera for each image and the corresponding images to improve details of the scene and build a dense point cloud. Then, the mesh model of the scene is generated in order to visualize better the object and detect cracks. The textured model is built in the end. This process is organized according to Figure 1.

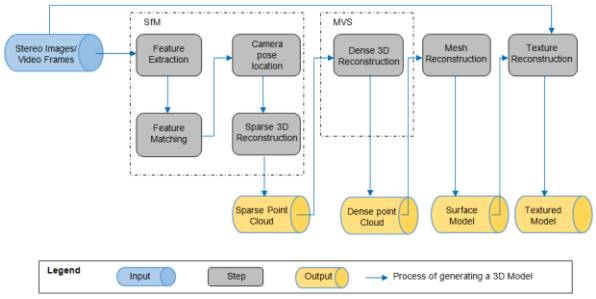

Figure 1: General framework of a 3D reconstruction

The images taken on the fields of Fravízel company were close range and in an uncontrolled open environment with similar conditions to a Quarry mine. The data acquisition needs to consider a series of rules, according to software requisitions. In terms of object/scene requirements, not textured, shiny, highly reflective or transparent objects should be avoided, as well as moving objects within the scene, but also flat objects [20]. A good capture of photos should avoid direct light such as in the daylight, shadows and one colored surfaces [21]. In terms of capturing scenarios, it is better to have more photos than not enough, the object of interest should take the maximum area in the scene, flash should be avoided and good lighting is crucial to achieve better quality of results [20]. The images must have a side overlap of, at least, 60% and frontal overlap of 80%, without moving the object or scene [21]. In this regard, a video was recorded at the same time from both cameras and there were 68 images recovered. These images were taken in pairs, and so the 1st image and the 35th image were taken at the same time but with the left camera and the right camera respectively, while the limestones still and the cameras facing the scene moving, with an angle of 83.029º.

The adopted cameras were Genie Nano C4020 Color, which is a CMOS sensor of 12,4 MP and their respective lenses of 83,029º of field of view [22]. This camera can be represented as a pinhole camera model to portray the fixed internal camera parameters and establish the mathematical relation between the camera coordinate and the pixel coordinate in the image frame. The internal and external parameters of the cameras applied in this study were calculated in [22].

The hardware used for this thesis is a computer MSI GT72 3QE Dominator Pro. The most important specifications for a reconstruction are the CPU, RAM and GPU. The laptop MSI GT72 3QE Dominator Pro has CPU of the 5th generation Intel® Core™ i7 5950HQ / 5700HQ processor, a 32 GB RAM and a NVIDIA GeForce GTX 980M. The most demanding pipelines are the commercial ones, therefore the minimal hardware requirement configuration was established according *Agisoft Metashape* [20]. The minimal requirements, 4 GB of RAM, can build a model based on 30 to 50 photos of a resolution of 10 MP. However a 16 GB RAM can process up to 300-400 photographs [20]. A necessary characteristic is that the GPU NVIDIA should be supported by CUDA in order to accomplish reconstruction in certain pipelines such as COLMAP, *Meshroom* and *Agisoft Metashape* [23][20][21]. For this project CUDA was installed for the pipelines to utilize the processing power of CUDA. In terms of software most of the popular software work on Windows XP or later (32 or 64 bit), Mac OS X Mountain Lion or alter, Debian/Ubuntu. This assignment was executed in Windows 10.

This study focuses in the few most popular open source and commercial software. Along with the most leading software of 3D reconstruction, *VisualSFM* and *COLMAP* generated the best results according to [24], therefore these pipelines were chosen to be analyzed. MeshLab is a well-known open-source system for processing and editing 3D meshes but also commonly used to open the models created by other pipelines such as *COLMAP*, that can compute a 3D mesh but not illustrate it. *Meshroom* was selected among other pipelines for being a free, open-source 3D reconstruction software based on the commercial *AliceVision* framework and thus providing quality and robustness to reconstruct, not only sparse and dense point clouds, but also, surface and textured models. Contrarily to other 3D reconstruction pipelines, *Meshroom* developers have been recently improving its contents with a major update in 2019 contributing to a better quality and speed in generating 3D models, while competing with some commercial pipelines [21], Agisoft Metashape was also chosen for being an acclaimed 3D reconstruction pipeline and for its recent update transitioning from the popular *Agisoft Photoscan* to *Agisoft Metashpe* while improving performance. Therefore, the open source pipelines focused in this study are the following: *VisualSFM* [25], *COLMAP* [10][26][27], *Meshroom* [21] and MeshLab. The commercial pipeline used is the *Agisoft Metashape* [20].

Most of the algorithms implemented on the open-source pipelines are known, therefore it was chosen a standard sequence of methods for all software to achieve a 3D model for proper comparison. SIFT was the feature detection approach for all pipelines, even Agisoft applies a variation of SIFT. The features were matched through brute force in an exhaustive approach for all open-source pipelines. The method PnP followed by bundle adjustment were executed to estimate and correct the camera poses,

respectively. All open-source pipelines adopt RANSAC for outlier removal. *VisualSFM* and *COLMAP* use CMVS/PMVS to produce a Dense point cloud. On the other hand, *Meshroom* applies SGM. *VisualSFM* does not generate surface models, yet, *COLMAP* produces 3D meshes through two different methods: Screened Poisson surface reconstruction and Delaunay triangulation based surface reconstruction. *MeshLab* also implements Screened Poisson surface reconstruction and *Meshroom* achieves a surface model through the Delaunay triangulation-based surface reconstruction.

Nevertheless, some software weren't able to generate 3D models based on the frames obtained in the open pit mine. In order to solve this issue, it was implemented 5 preprocessing methods to enhance the images.

Image Preprocessing

The uncontrolled environment of the open pit mine retrieves images with inconsistent lighting and shadows between the two cameras. The brightness and contrast of an image can change how the images are processed in a 3D reconstruction pipeline, by varying the number of detected features and matched points, but also, compromise the estimation of the camera pose location. Between both cameras the lighting should be similar for the software to be able to detect features and consider those as the same scene but also the enhancement of brightness and contrast allows the software to detect more features and therefore tie points. Since the output information from the SFM process is crucial to apply MVS and build a surface reconstruction, the higher quality is the camera parameters, their pose location and the overall points of the sparse point cloud, the more correct and authentic is the dense point cloud and the surface model. As it is shown in Figure 2, the images from the first three rows and a half are the camera on the left and the rest of the images are the camera on the right. At the time of the day that the images were captured, the right camera retrieves images brighter than the left camera.

Figure 2: Original images

There were 6 methods applied to these images to improve their quality in order to build a good 3D model. The preprocessing methods were the following: An averaging method, histogram equalization, CLAHE, gamma correction, gamma correction with histogram equalization and gamma correction with CLAHE. These methods were all implemented in *python* through two modules, the *Pillow* and *OpenCV*.

Averaging Method

The averaging method aimed to compute the perceived brightness and contrast to all images and average them. Brightness is the overall lightness and darkness of the image and contrast is the difference in brightness between regions of an image. The perceived brightness of the original images was calculated through the arithmetic mean of each color band in the image (RGB values). Considering RGB color space as a cube where each of the three colors are an axis, a corner of this cube corresponds to black with RGB (0,0,0) and the opposite corner to white with RGB (255,255,255). And so, the perceived brightness is the 3D distance between the first corner to the actual color value for each band, which in this case is the mean of the RGB values of the image. Also, it is necessary to take into account that some colors look brighter than others by giving a different weight to each axis, as it is shown in equation 0.1.

$$Brightness = \sqrt{0.241R^2 + 0.691G^2 + 0.068B^2} \quad 0.1$$

The perceived contrast can be calculated by converting the image into a CIELAB color space that expresses three values: L for lightness, A from green to red and B from blue to yellow. Once the lightness values for each pixel of the image are computed, two images are created with the maximum and minimum values through dilation and erosion respectively. A 5 by 5 Mask is run through each pixel and replacing its value with the maximum or minimum value within the mask for dilation and erosion respectively. The contrast of the image is calculated by subtracting the eroded image from the dilated image and dividing its sum. Then, the average of the contrast matrix of each original image is computed, according to equation 0.2.

$$average\ contrast = \frac{\sum_{i,j=0}^{i,j=m,n} \left(\frac{max - min}{max + \min}\right)_{i,j}}{m \times n} \quad 0.2$$

where max and min are $m \times n$ matrices of the dilated and eroded images respectively.

Once the perceived brightness and contrast are known, it is calculated the average of the brightness and contrast of the image from the camera in the left and right. In order to have images that can be perceived as the image from the same scene and smooth the light of all images in order to have images with similar brightness and contrast it was applied an interpolation between the original images and an image with only black pixels with same size as the first image. This interpolation is implemented according to a certain factor. For the images on the right and on the left, in the case of brightness, this factor is the percentage of the absolute value of the difference between the overall average brightness and the average brightness of the right (Camera 1) and on the left images (Camera 2), respectively. In the case of contrast, the images on the right and on the left, this factor is the percentage of the absolute value of the difference between the overall contrast and the average contrast of the right (Camera 1) and on the left images (Camera 2), respectively. This factor certifies that the interpolation is between $(1 - factor\ \%)$ of the black images and $(factor\ \%)$ of the original images. This method is able to

ensure that all images tend to the same overall average brightness and the same overall average contrast. Nevertheless, the outliers continue to be outliers and would need to be removed. Further, even though the images have the same average perceived brightness and contrast, the images continue to have varying lightness value pixels within the images.

Histogram Equalization

Histogram equalization is an image processing technique of contrast enhancement. The histogram represents graphically the intensity distribution of an image with pixel values in X-axis (ranging between 0 to 255) and the corresponding number of pixels in the image on the Y-axis. Therefore, histogram equalization stretches the intensity range of the images to produce a good image with pixels of different intensity values instead of having peaks, concentration of pixels with small range of intensity value.

Histogram Equalization aims to stretch the intensity range of these images by calculating the cumulative distribution function (CDF). Once the cumulative distribution function is known, it is applied a transform onto the original image to produce an image with a flat histogram through a linearized distribution function along with the intensity range of these images by multiplying the cumulative probability of each pixel intensity by the range of the intensity that wants to be stretch which in this case is 256 intensity values. Finally, the decimal values obtained through these calculations are rounded to their lower integer value.

The results of histogram equalization on the original images demonstrates that the histograms of the images from the left camera were flattened, and the whiter pixels decreased. The histograms of the images from the right camera show that the histogram equalization eliminated the main peak of darker pixels.

CLAHE

CLAHE or contrast limited adaptive histogram equalization is another image processing technique to enhance contrast. This approach is more robust than normal histogram equalization for being an adaptive histogram equalization. The process lies in computing several histograms to different sections of the image in order to redistribute the intensity values of the image. It also avoids the possible noise caused by over amplification by applying a contrast limiting method to each neighborhood from where the transformation was established.

The procedure starts by splitting the image into R, G and B, and for each image channel, the images are divided into small blocks of 8 by 8 elements and applying histogram equalization on them. If this small area has noise, this noise will be amplified, therefore to improve this adaptive histogram equalization, it is applied a contrast limit by establishing that, if any histogram bin is above a certain threshold (in this case 40 pixel intensity value) than those pixels are distributed uniformly to other bins before applying histogram equalization [6]. Afterwards, the histogram equalization is applied in the same way as it was described previously in this project. Lastly, the R, G and B output for each image are merged

The results of CLAHE implementation on the original images show that the histograms from the left images have less darker and whiter pixels and the darker peak of pixels from the right images decreased significantly.

Gamma Correction

Gamma correction is another image processing technique that enhances the contrast and brightness of an image through a nonlinear transformation between the input and the output values. This method is also known as the Power Law Transform and it can be written according to equation 0.3 [6]. The value of gamma γ is changed to choose its best value in order to produce the best output image. When $\gamma < 1$ the dark regions of the original image get brighter and the histogram is shifted to the right and when $\gamma > 1$ the original brighter regions turn darker and the histogram is shifted to the left.

$$O = \left(\frac{I}{255}\right)^{\gamma} \times 255 \qquad 0.3$$

where γ is the gamma value, O is the output image values, I is the input image values for an image with values between 0 and 255.

Gamma adjustment can be implemented by creating a table with 256 elements computing the equation 0.3 where I is a number that ranges between 0 and 255 and gamma is a given variable. The adjusted image is obtained by applying a gamma and a lookup tables (LUTs) function, from *openCV*, where it is performed a transformation that assigns a new pixel value to each pixel of the source image, according to the table previously created. This method is less computationally expensive than computing the equation 0.3 for each 12000 pixels per image.

Based on the images with gamma correction and analyzing the results through their histograms, the chosen gamma correction was $\gamma = 0{,}4$ for both left and right images. The implementation of Gamma correction shows that the histograms from all images shifted the darker pixel peak to a more even tone, around 127 pixel value.

Gamma correction with Histogram equalization

Gamma correction with Histogram equalization was another implemented method with the objective of stretching the intensity range of images that already have been gamma corrected to have similar brightness. This method is the union of the 2 approaches already explained. First, a gamma correction of $\gamma = 0.4$ was applied to all images and then a histogram equalization was implemented to all corrected images. This implementation showed the distribution most pixels in the darker and whiter pixel peaks, decreasing the images brightness but also contrast.

Gamma correction with CLAHE

Similarly to the concept of gamma correction with histogram equalization, Gamma correction and CLAHE is the association of methods previously explained. However, in this case, a CLAHE is applied on images that have already been gamma corrected to have similar brightness. This CLAHE implementation is identical to the

previous approach, implying that the histogram equalization is enforced on small blocks of 8 by 8 elements and the contrast limit is 40 pixel value. This implementation showed a decrease in darker pixels to more grey pixels due to gamma correction and a small decrease in the whiter pixel peak

Preprocessing method comparison

The different approaches previously implemented are compared in this section in order to sort out the best input images to build a 3D model. The comparison of the 6 preprocessing methods is done by processing the images from each implementation through a 3D reconstruction software. The software chosen to reach to the best preprocessing method and its output images was COLMAP for being one of the best open source software for SFM implementation [24][23] .The best preprocessed images are the original images that were subjected by a gamma correction followed by a histogram equalization. Gamma correction with histogram equalization was the second preprocess with more sparse points produced and the third fastest, 8.8% faster than the gamma correction with CLAHE that built a cloud with 1.3% more sparse points. As shown in Figure 3, the first image was completely recovered from the dark tone original image, seen in Figure 2. Further, the difference in brightness and contrast between the images from the camera on the left and on the right are almost unidentifiable. This set of preprocessed images are going to be adopted for the rest of this study in order to reach to the best results.

Figure 3: Preprocessed images through Gamma correction and Histogram equalization

Results and Discussion

For each output stage of the reconstruction the best preprocessed set of images is introduced on each software to further analyze the models. The results are divided in 4 stages: The first stage compares different software, the second stage determines the best algorithm combination within the previously concluded best software, the third stage achieves a 3D model with known intrinsic parameters for comparison as well as the fourth stage that accomplish a 3D model with known intrinsic and extrinsic parameters

Stage 1

This first stage compares the different types of software to conclude the most suitable pipeline in order to reconstruct 3D models of limestones. For all open-source software, it was used similar procedures with the objective of comparing each pipeline The four output stages considered were: Sparse point Reconstruction, Dense point Reconstruction, Surface Reconstruction and Textured Reconstruction.

Sparse point Cloud

The duration of the sparse point cloud generation, *VisualSFM* achieved the fastest feature detection of the 4 pipelines analyzed, however, *VisualSFM* is an ineffective software at producing a tie point cloud due to the misestimation of the camera poses, and therefore, nonviable to continue the analysis. Overall, *Meshroom* was the fastest software by taking 46.6% less time than *COLMAP* and 58.8% less than *Agisoft*. *COLMAP* accomplished a 22.8% faster performance compared to Agisoft Metashape. Although, *VisualSFM* proved to be nonviable, this pipeline was able to detect more features than *Meshroom* since the miscalculation only occur on the last step of SFM. *COLMAP* detected 1.93 times more features than *Meshroom* but only 14% features compared to *Agisoft Metashape*. Similarly to the number of features detected, Agisoft Metashape produced a cloud with distinctly more points than the other pipelines, creating a sparse point cloud with 5.5 times more tie points than *COLMAP* and 8.5 times more than *Meshroom*.

Dense Point Cloud

Agisoft Metashape achieves an even more complete dense reconstruction compared to *COLMAP*. It is also evident the need for more information in order to build the posterior of the limestones and the front of the second stone. Another apparent detail is the focus of dense points only on the relevant forefront of the limestone line, eliminating all the information from the background, contrarily to *COLMAP*. Since Meshroom software does not display details of the dense point cloud generation, it was not possible to know the total time of this step and the number of dense points. Therefore, *Meshroom* performance can only be compared in the other stages of reconstruction. Nevertheless, it was concluded that *COLMAP* was 3.5 times slower and calculated 60% less points than *Agisoft Metashape*.

Surface Model

COLMAP's software can compute surface reconstructions but its graphical user interface is unable to show surface models. There are two methods that can be executed in *COLMAP*: Screened Poisson surface reconstruction and Delaunay triangulation-based surface reconstruction. The surface model generated based on the Poisson reconstruction displayed an undesirable model with acceptable structure but a substantially amount of noise. On the other hand, Surface reconstruction based on Delaunay triangulation retrieved a model more accurate and less noisy, but still unusable due to the imprecision of the structure compared to the real limestones. *MeshLab* can create surfaces based on dense point cloud through the process of Screened Poisson surface reconstruction. MeshLab's input was the dense point cloud produced by *COLMAP*. The extent of noise present in the dense point cloud from *COLMAP* was too large to generate a suitable

model, creating an unusable 3D mesh that is even less akin to the ground truth than the model built by *COLMAP* with identical method. *Meshroom* was able to produce an acceptable 3D mesh of the scene since most of the noise was filtered by the software, achieving a 3D model of the limestone identical to the ground truth. Contrarily to the other open-source software, the holes in the back of the stones were not closed, therefore, the formation of a structure on the back that is a misrepresentation of the real limestones was prevented but the opening on the front of both limestones in line was its consequence. *Agisfot Metashape* generated a 3D surface identical to the 3D mesh created by Meshroom, regarding the nonexistence of background noise and the stone's geometry. Agisoft's model has less clutter compared to Meshroom and the openings on the limestones are closed, however, this built a distortion on the back of each limestone, but the front openings were closed.

Despite the short elapsed time of a Poisson reconstruction, it is evident that both *COLMAP* and *MeshLab* could not produce an applicable 3D mesh for Quarry industry use, based on the number of faces produced by this method and the images of the 3D surfaces. The Delaunay triangulation-based surface reconstruction implemented by *Meshroom* and *COLMAP* had similar number of faces and duration time. However, contrarily to Meshroom, most of the faces generated by *COLMAP* are clatter from the background and, even though, it created a better model compared to the Poisson reconstruction from *COLMAP* and MeshLab, it still has severely less detail compared to *Meshroom* and *Agisoft Metashape*. *Agisoft Metashape* achieved a 3D surface of the quarry stone line 70% faster and a surface with 1.3 times more faces than *Meshroom*. Therefore, the open source *Meshroom* accomplished almost the same amount of detail compared to the commercial pipeline, *Agisoft Metashape*, since there are no faces to close the openings on the surfaces.

Textured Model

The texture model reconstruction is the last phase of a 3D reconstruction, although the 3D mesh of the limestone can be sufficient to identify the proper regions of cutting stone, textured model adds information for an exact decision. *COLMAP* does not create a texture file for the mesh, it only uses the vertex color to apply texture, therefore it isn't compared to the other pipelines.

Meshroom and *Agisoft Metashape* improved the 3D surface reconstructed, through texture implementation. While it was applied texture on the clutter in *Meshroom* and the opening stayed the same, as shown in Figure 4, *Agisoft Metashape* developed a model that textures most of the closing structure from the backside and frontside of the stones. Yet, *Agisoft* was able to perform a texture reconstruction over 2 times faster than *Meshroom*.

Figure 4: Textured reconstruction of limestones by Meshroom

Stage 2

Meshroom has different algorithms to perform each cloud reconstruction. For dense, surface and textured reconstruction, this pipeline only has one method, therefore, the analysis to improve the current *Meshroom* reconstruction is focused in the SFM stage. *Meshroom* has different algorithms to perform feature detection, however, SIFT is the only one applied since it creates robust and accurate feature descriptors essential to produce a good reconstruction, but also, for the elapsed time being already negligible. Image matching and feature matching can be evaluated between various algorithm combination, but camera pose estimation and correction can only be accomplish by the same established methods. Therefore, the image and feature matching algorithms are compared according three image matching and three feature matching methods combined: The exhaustive, vocabulary tree and sequential methods for images matching and the brute force, ANN and Cascade Hashing for feature matching. Any combination built a sparse cloud with similar durations and number of sparse points, however, the method cascade hashing produced worse results in terms of number of tie points combined with any image matching process. The number of tie points is the most valuable aspect to build a 3D model in relation to such small differences in time, therefore the best matching solution was feature matching through brute force in an exhaustive image matching technique.

Stage 3

A correctly scaled 3D model can be built if the intrinsic parameters are precise and known. In order to generate a scaled model, the same set of preprocessed images were provided as input to *Meshroom* software, as well as three different types of intrinsic for comparison analysis: The intrinsic parameters without distortion coefficients and the intrinsic parameters with distortion coefficients by [22] and the intrinsic parameters generated by *Meshroom* in the previously concluded best sequence of matching algorithms.. The number of tie points, the time to build a sparse cloud and the overall geometry of the model were the analyzed components.

The best apparent sparse point cloud was generated by the intrinsic parameters produced by *Meshroom* with faster and a greater number of tie points. The model built from the intrinsic with distortion parameters is a disorganized cloud of sparse points and the model based on the intrinsic without distortion parameters produced a cloud with tie points captured from the right camera unaligned with the tie points originated by

the images from the left. Subsequently, the intrinsic calculated by [22] proved to be unviable and the only possible application is the intrinsic by *Meshroom*, based on the distribution of the tie point in the 3D model. By giving the intrinsic previously calculated, the number of tie points increased an insignificant amount, however, the elapsed time to generate a sparse point cloud was 10% faster.

Stage 4

The final study to improve the 3D model generation quality and speed is to input the intrinsic and the extrinsic parameters along with the images in the software. From the previous analysis it was determined that only the intrinsic parameters computed by *Meshroom* are viable and therefore it is the only intrinsic used on this last analysis. The extrinsic parameters obtained by [22] were compared to the parameters computed by *Meshroom* in the same condition of the intrinsic comparison analysis. The number of tie points, the time to build a sparse cloud and the overall geometry of the model were the analyzed components.

The intrinsic and extrinsic parameters computed by *Meshroom* generated more tie points and was slightly faster than the parameters obtained by [22]. The intrinsic parameters calculated by *Meshroom* with the extrinsic parameters obtained by [22] produced a sparse cloud with unaligned points from the left and right cameras introducing an unapplicable option. While the 3D model obtained by the parameters computed in *Meshroom* has a geometry similar to the real object. However, by giving he intrinsic and extrinsic parameter previously computed by *Meshroom* the number of tie point decreased 22%, while the elapsed time was 33% faster and the model still looks similar to the ground truth, as shown in Figure 5.

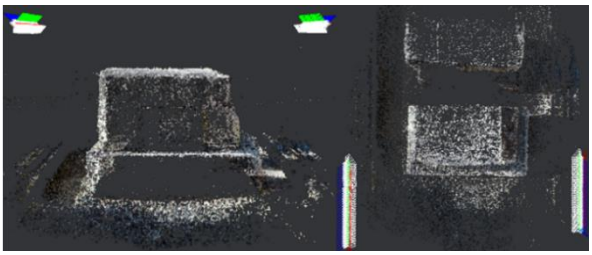

Figure 5: Sparse Cloud given intrinsic and extrinsic parameters by Meshroom

Conclusion and Future Work

Meshroom and *Agisoft* achieved the best performances. *Meshroom* generated a 3D textured model with 1,428,422 faces in 59.71 minutes while *Agisoft Metashape* took 84.727 minutes to compute the same model with 1,989,573 faces. The detail between both models has meaningless variations, the most apparent was the *Agisoft* attempt to close the openings on the meshes, creating a distortion on the back of the limestone, while *Meshroom* has openings in the stones. However, *Meshroom* was 30% faster than *Agisoft Metashape*. Since *Meshroom* enables the user to choose each method used in every step, it was possible to iteratively try different algorithms and conclude the fastest and most complete output. The combination of exhaustive image matching with brute-force feature matching confirmed to be the fastest and most complete to generate a sparse cloud. Further, the intrinsic and extrinsic parameters were introduced to build the sparse point cloud in order to achieve the fastest generation of a model without compromising the model detail, based on the concluded sequence of best matching algorithms. Before introducing the extrinsic, different versions of intrinsic parameters were compared, but only the intrinsic parameters computed by Meshroom were viable, proving to decrease the sparse point cloud generation by 10%. Since only the intrinsic parameters computed by *Meshroom* are applicable, these intrinsic parameters were introduced with the extrinsic concluded in [22] and along with the extrinsic previously computed by *Meshroom* for comparison. This final analysis determined that the extrinsic parameters in [22] weren't fit to produce a model with geometry similar to the real limestones while the parameter computed by *Meshroom* achieved a similar cloud structure with less 22% tie points in less 33% time than the reconstruction without the intrinsic and extrinsic parameters. Overall, the preprocess of the images captured in an uncontrolled environment proved to be crucial to build a 3D model based on open source or commercial software, further, the input of the intrinsic and extrinsic parameters to improve the time duration of a scaled sparse point cloud reconstruction. It was concluded that it is possible to reconstruct a 3D model of the limestone in an uncontrolled environment through *Meshroom* in 55.25 minutes.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisor, Professor Jorge Martins for his guidance and support during the development of this project. I would also like to acknowledge the company Fravízel-Equipamentos Metalomecânicos, SA in the e-techStone 4.0 project for being accessible and helpful during this process.

References

[1] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Second Edi. 2004.

[2] L. G. Roberts, "Machine perception of three-dimensional solids." 1963, [Online]. Available: http://dspace.mit.edu/handle/1721.1/11589.

[3] S. El Hazzat, A. Saaidi, and K. Satori, "Euclidean 3D reconstruction of unknown objects from multiple images," *J. Emerg. Technol. Web Intell.*, vol. 6, no. 1, pp. 59–63, 2014, doi: 10.4304/jetwi.6.1.59-63.

[4] E. Vural and A. A. Alatan, "Outlier removal for sparse 3D reconstruction from video," *2008 3DTV-Conference True Vis. - Capture, Transm. Disp. 3D Video, 3DTV-CON 2008 Proc.*, pp. 341–344, 2008, doi: 10.1109/3DTV.2008.4547878.

[5] Y. Furukawa and C. Hernández, *Multi-View Stereo : A Tutorial*. .

[6] "OpenCV." https://opencv.org/ (accessed Oct. 08, 2020).

[7] Z. Al-Ameen, G. Sulong, A. Rehman, A. Al-Dhelaan, T. Saba, and M. Al-Rodhaan, "An innovative technique for contrast enhancement of computed tomography images using normalized gamma-corrected contrast-limited adaptive histogram equalization," *EURASIP J. Adv. Signal Process.*, vol. 2015, no. 1, pp. 1–12, 2015, doi: 10.1186/s13634-015-0214-1.

[8] D. G. Low, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, pp. 91–110, 2004, [Online]. Available: https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf.

[9] J. Cheng, C. Leng, J. Wu, H. Cui, and H. Lu, "Fast and accurate image matching with cascade hashing for 3D reconstruction," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 1–8, 2014, doi: 10.1109/CVPR.2014.8.

[10] J. L. Schonberger, "COLMAP — COLMAP 3.7 documentation." https://colmap.github.io/ (accessed Jan. 28, 2021).

[11] M. A. Fischler and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Graph. Image Process.*, vol. 24, no. 6, pp. 381–395, 1981, doi: 10.1145/358669.358692.

[12] D. Nistér, "An efficient solution to the five-point relative pose problem," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 6, pp. 756–770, 2004, doi: 10.1109/TPAMI.2004.17.

[13] R. I. Hartley, "In Defense of the Eight-Point Algorithm," *EEE Trans. PATTERN Anal. Mach. Intell.*, vol. 19, no. 6, pp. 580–593, 1996.

[14] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz, "Multicore bundle adjustment," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, no. 1, pp. 3057–3064, 2011, doi: 10.1109/CVPR.2011.5995552.

[15] H. Hirschmüller, "Stereo processing by semiglobal matching and mutual information," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 328–341, 2008, doi: 10.1109/TPAMI.2007.1166.

[16] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multiview stereopsis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 8, pp. 1362–1376, 2010, doi: 10.1109/TPAMI.2009.161.

[17] M. Kazhdan and H. Hoppe, "Screened poisson surface reconstruction," *ACM Trans. Graph.*, vol. 32, no. 3, pp. 1–13, 2013, doi: 10.1145/2487228.2487237.

[18] H. K. Zhao, S. Osher, and R. Fedkiw, "Fast surface reconstruction using the level set method," *Proc. - IEEE Work. Var. Lev. Set Methods Comput. Vision, VLSM 2001*, pp. 194–199, 2001, doi: 10.1109/VLSM.2001.938900.

[19] M. Callieri, G. Ranzuglia, M. Dellepiane, P. Cignoni, and R. Scopigno, "Meshlab as a Complete Open Tool for the Integration of Photos and Colour with High- Resolution 3D Geometry Data Marco," *Comput. Appl. Quant. Methods Archaeol. 1990*, no. 565, 2011.

[20] Agisoft, "Agisoft Metashape User Manual," p. 160, 2020, [Online]. Available: https://www.agisoft.com/pdf/metashape-pro_1_5_en.pdf.

[21] "Meshroom Manual — Meshroom v2020.1.0 documentation." https://meshroom-manual.readthedocs.io/en/latest/ (accessed Nov. 12, 2020).

[22] F. F. Monteiro, "Integration of an optical system for 3D reconstruction," *Master thesis Sci. degree Mech. Eng.*, no. January, p. Instituto Superiror Técnico, 2021.

[23] J. L. Schonberger and J. M. Frahm, "Structure-from-Motion Revisited," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 4104–4113, 2016, doi: 10.1109/CVPR.2016.445.

[24] S. Bianco, G. Ciocca, and D. Marelli, "Evaluating the performance of structure from motion pipelines," *J. Imaging*, vol. 4, no. 8, pp. 1–18, 2018, doi: 10.3390/jimaging4080098.

[25] "VisualSFM : A Visual Structure from Motion System." http://ccwu.me/vsfm/ (accessed Oct. 05, 2020).

[26] Y. Chen, Y. Chen, and G. Wang, "Bundle Adjustment Revisited," 2019, [Online]. Available: http://arxiv.org/abs/1912.03858.

[27] J. L. Schönberger, E. Zheng, J. M. Frahm, and M. Pollefeys, "Pixelwise view selection for unstructured multi-view stereo," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9907 LNCS, pp. 501–518, 2016, doi: 10.1007/978-3-319-46487-9_31.