KLE Society's
KLE Technological University, Hubballi.

A Minor Project Report

On

Generation of 3D Model for IHDS Dataset Using 2D Images

submitted in partial fulfillment of the requirement for the degree of

Bachelor of Engineering

In

School of Computer Science and Engineering

Submitted By

Karnika Bhardwaj	01FE18BCS085
Sakshi Jha	01FE18BCS182
Sanjay Keshwar	01FE18BCS192
Savitri Khyadad	01FE18BCS196

Under the guidance of
Dr. Meena S M

SCHOOL OF COMPUTER SCIENCE & ENGINEERING

HUBBALLI – 580 031

Academic year 2020-21

KLE Society's
KLE Technological University, Hubballi.

2020 - 2021

SCHOOL OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that Minor Project entitled Generation of 3D Model for IHDS Dataset Using 2D Images is a bonafied work carried out by the student team Karnika Bhardwaj USN: 01FE18BCS085, Sakshi Jha USN:01FE18BCS182, Sanjay Keshwar USN:01FE18BCS192, Savitri Khyadad USN:01FE18BCS196, in partial fulfillment of completion of Sixth semester B. E. in Computer Science and Engineering during the year 2020-2021. The project report has been approved as it satisfies the academic requirement with respect to the project work prescribed for the above said program.

Guide	Head, SoCSE
Dr. Meena S. M	Dr. Meena S. M

External Viva -Voce:

Name of the Examiners	Signature with date
1.	
2.	

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of a number of individuals whose professional guidance and encouragement helped me in the successful completion of this report work.

We take this opportunity to thank Dr. Ashok Shettar, Vice-chancellor and to Dr. N H Ayachit, Registrar, KLE Technological University, Hubballi.

We also take this opportunity to thank Dr. Meena S M, Professor and Head of Department, Department of Computer Science and Engineering for having provided us academic environment which nurtured our practical skills contributing to the success of our project.

We sincerely thank our guide Dr. Meena S M, Professor and Head of Department, Department of Computer Science and Engineering for her guidance, inspiration and wholehearted co-operation during the course of completion.

We sincerely thank the reviewers for their suggestions, project co-ordinator Dr. Sujatha C. and faculty incharges for their support, inspiration and wholehearted co-operation during the course of completion.

Our gratitude will not be complete without thanking the Almighty God, our beloved parents, our seniors and our friends who have been a constant source of blessings and aspirations.

Karnika Bhardwaj - 01FE18BCS085
Sakshi Jha - 01FE18BCS182
Sanjay Keshwar - 01FE18BCS192
Savitri Khyadad - 01FE18BCS196

ABSTRACT

Many applications, such as architectural heritage preservation, city-scale modeling, and so on, require the extraction of 3D structures from 2D images. Various techniques for reconstructing the three-dimensional shape and appearance of objects have been developed. In this field, progress has been rapid over the last two decades. We now have proven methods for creating a 3D representation of a scene from hundreds of overlapping photos. We can construct detailed dense 3-D three-dimensional surface models using stereo matching if we have enough sets of views of a particular object. To create the 3D models, we use open-source image-based 3D-reconstruction pipelines. External camera poses per picture and 3D point clouds are calculated utilizing camera intrinsic characteristics and feature matching with the aid of the pipelines. After computing the camera positions, a patch-based stereo method is used to do dense reconstruction for large-scale scenes. However, because of occlusion, scanning angle, reflectance, and raw data preprocessing, some point data is invariably lost, resulting in holes in the reconstruction surface, making it unfit for a variety of applications. As a result, hole filling is a critical phase in the project if accurate 3D models are to be produced. To fill holes, we use mesh-based hole filling techniques. To improve the 3D models, we utilize the pymeshfix python package. The 3D models were obtained for the IHDS dataset using structure from motion and multi-view stereo.The models that resulted had some holes. Using the pymeshfix python package, the holes were identified and repaired, resulting in accurate 3D models.

Keywords : *3D reconstruction, structure from motion, multi view stereo, hole filling.*

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Reconstruction of 3D geometry is one of the most fascinating fields in computer vision because of its large application.3D reconstruction is the technique of capturing the shape and look of real things in computer vision and computer graphics. This procedure can be carried out in either an active or passive manner.2D to 3D reconstruction is a technique of transforming a set of 2D images into a 3D form. Calculating the depth is the most important factor while converting 2D images to a 3D form because the main difference between 2D and 3Dimages in the presence of depth in the 3D images. 3D image reconstruction has proven to be very important in various fields like entertainment, augmented reality, animation, and robotics.3D image generation allows one to obtain certain features that cannot be obtained from a 2D image like volume or relative position of other scenes. Many easy applications based on Structure from Motion (SfM) now allow users to generate high-quality 3D models on their smartphones without any prior expertise or understanding of the method. The project is to generate detailed 3D models from the IHDS temple dataset images alone. The importance of 3D reconstruction of scenes has led to the development of different tools and methods likeCAD, active methods, and passive methods. The passive methods which use images are widely used because of the fast way of reconstruction. The passive methods(Multi-view Stereo Vision(MVS))are used to generate the 3D models in the project. Our project is based on recent work on structure from motion that recovers camera parameters, estimation of poses, and generates a3D point cloud from the image dataset. The intrinsic and extrinsic parameters of the camera are calculated from the images using certain computer vision techniques rather than relying on camera or any other equipment

1.1 Motivation

Structure from Motion photogrammetry with multi-view stereo provides detailed 3D models using a set of 2D images captured from different digital cameras and at different views. This method provides a 3D dense point cloud that is accurate when compared to those generated by laser scanning methods at a much less expensive. Passive methods of obtaining a 3D model are advantageous over the active methods, because they can create 3D models of objects or buildings which no longer exist, but can be seen only through images and recordings. A cheap digital camera embedded with a 3D reconstruction algorithm can be a good alternative to

various other expensive equipment. The existing solutions for 3D reconstruction are not satisfactory. Techniques like 3D scanning, terrestrial laser scanning, and airborne laser scanning are expensive and are limited by equipment portability and terrain roughness that can cause loss of data. Sfm can perform well in such scenarios. Different types of digital cameras can be used, including smartphones. Structure-from-Motion Photogrammetry is a low-cost technique to digitalize historical documents or monuments.

1.2 Literature Review

The important methods in the 3D reconstruction include Sfm and MVS. Structure from Motion (SfM)[1] is a technique for reconstructing the 3-dimensional structure of a scene or object using a sequence of 2-dimensional pictures. It's a topic of research in the fields of computer vision. SfM refers to the capability of humans to restore three-dimensional structure from a projected 2D motion field of a mobile scene in biological terms.The difficulty of determining structure from motion is comparable to that of determining structure through stereo vision. The relationship between pictures and the reconstruction of a 3D object must be discovered in both cases.Features are tracked from one picture to the next to identify correlation between them.The feature detector used is SIFT. As features, it uses the maxima from a difference-of-Gaussians (DOG) pyramid. Finding a dominating gradient direction is the first step in SIFT. The description is rotated to suit this orientation to make it rotation-invariant.[2]. The identified characteristics from all of the pictures will then be compared. Some of the matching characteristics are sometimes wrongly matched. This is why RANSAC[3] should be used to filter the matches. The feature trajectories over time are then utilised to rebuild the 3D locations of the features and the motion of the camera. Structure from motion may be approached in a variety of ways. Camera postures are solved for and added to the collection one by one in incremental SFM[4]. The postures of all cameras are solved at the same time in global SFM.

Stereo pair selection, depth-map computation, depth-map refining, and depth-map merging are the four phases of the MVS technique. Because patch-based stereo may create depth maps with tolerable errors, it can reconstruct pretty accurate and dense point clouds, which can then be further improved via the depth-map refining process. Each of the raw depth maps created is improved by stereo vision by consistency checking against their surrounding depth-maps, because they may include noises and mistakes. To achieve a final reconstruction, all of the improved depth maps are combined together.[5]

Structure from Motion and Multi View Stereo are two techniques that are be used to create 3D representations of objects, structures, and scenes. The algorithms' main flaws are that they are CPU and memory demanding.[6]

Three of the most popular open-source image-based 3D reconstruction processes are ex-

amined in this study.Photo Tourism[7] was one of the pioneers in the field, with several free and open-source solutions being accessible to the community. Based on conventional multiple view geometry concepts, OpenMVG provides a full SfM pipeline. SIFT and AKAZE are used for feature detection and description. Classic brute force, ANN-kD trees, and cascade hashing are used to match features. Geometric verification of image pairs is implemented using homography, essential or fundamental matrix. Sparse reconstruction is computed using incremental or global techniques, and then bundle adjustment is done using the Ceres solver. The OpenMVS library implements dense reconstruction for large-scale scenes using a patch-based stereo approach for this pipeline combination.[8]

The models generated using SfM and MVS may have holes due to less images available or poor textured images.Various hole filling techniques are proposed to fill the holes.[9] There are two types of current hole filling algorithms: point cloud-based approaches and mesh-based methods. Hole filling in this project is done using a mesh based hole filling algorithm.[10]

1.3 Problem Statement

Generation of 3D model for IHDS dataset using 2D images.The goal is to generate 3D models of the architectural heritage sites in IHDS dataset. The holes produced(if any) after the 3D model regeneration should be removed.

1.4 Applications

- A3D software is used in digital representation of many mechanical objects to build parts which will is further used to build and test these parts.
- Patients' lesion information may be shown in 3D on a computer, providing a fresh and accurate way to diagnose and therefore having critical clinical value.
- 3D reconstruction is used as technique to preserve various historical artifacts and architectural heritage conservation.

1.5 Objectives and Scope of the project

1.5.1 Objectives

- Data pre-processing for a crowd sourced data.
- Generation of 3D model from 2D images and removal of holes.

1.5.2 Scope of the project

The scope of the project include pavement engineering, medicine, robotic mapping, city planning, gaming, earth observation, archaeology.

Chapter 2

REQUIREMENT ANALYSIS

Requirement analysis specifies the software framework that will be created. It is based on the stakeholder's specification. The software specification sets out both practical and non-functional requirements, as well as list of use cases that explain how the software can communicate with the user for a perfect interaction

2.1 Functional Requirements

Functional requirements of a system are the system features and they focus on user requirements. Functional requirements describe the services that the system must offer. They help you capture the desired behavior of the system. They are basically what the system must do or must not do. The functional requirements of the proposed system are:

- User shall input the data.
- User shall be able to process and visualize the 3D model.

2.2 Non Functional Requirements

Non-functional requirements are essential as they serve as restrictions on the design of the system.They define the system attributes such as maintainability,scalability,security, reliability,usability, and performance.The non-functional requirements of the system are:

- The system should not crash while processing.
- The system should produce a detailed 3D model with minimum or no holes.

2.3 Hardware Requirements

The hardware requirements for the project are as follows:

- 16GB RAM preferable
- Core i5 processor

2.4 Software Requirements

The sofware requirements for the project are as follows:

- OpenMVG and OpenMVS Tools
- MeshLab
- CMake
- git
- C/C++ compiler like Visual Studio or GCC

Chapter 3

SYSTEM DESIGN

This chapter gives a brief description of system design details. The IHDS dataset contains 2D images that needs to be processed first. Some of the processes include Data clean-up, Data filtering and Classification. The dataset will be then ready for 3D reconstruction and 3D data processing. After all this processing the dataset can be 3D rendered and stored in structured data repository for future use in multiple application domains such as 3D model printing, Augmented Reality(A.R), etc.

3.1 Architecture Design

The 3D data processing consists of a number of steps such as Generation of sparse point cloud using Structure from Motion(SfM), Generation of dense point cloud from Multi-View Stereo(MVS) and finally Filling of Holes.

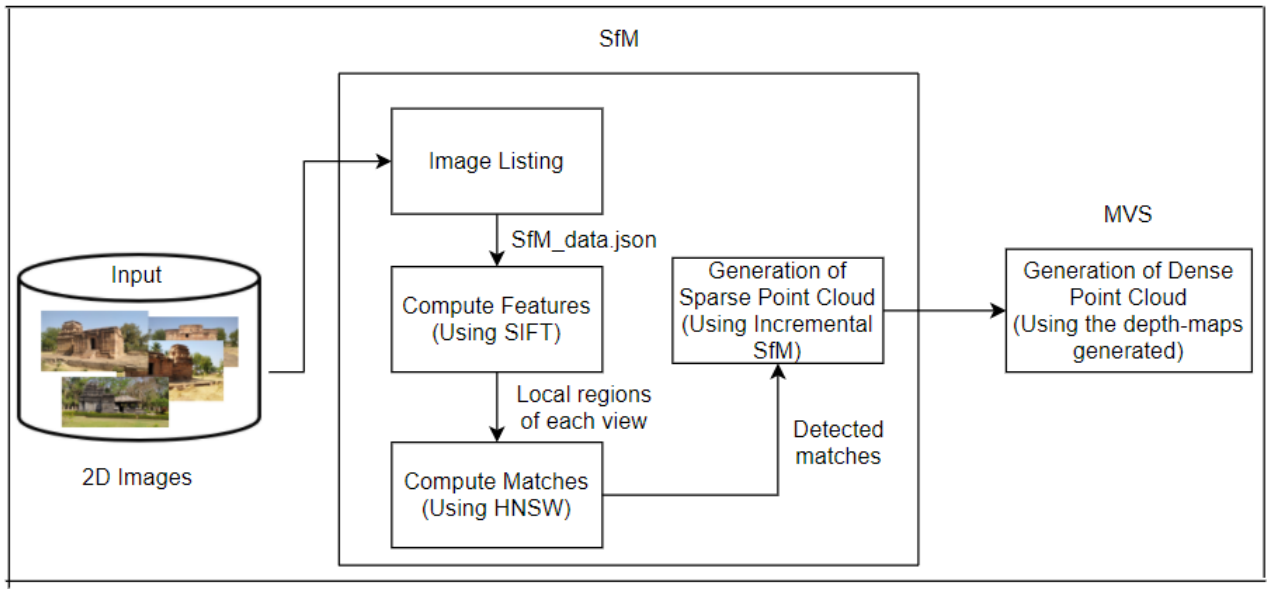

Figure 3.1: Proposed Methodology

Figure 3.1 shows the proposed methodology for the project. In SfM the 2D images are listed and the image details are converted in json format for further processes. SfM json

file is passed to another program where Scale-Invariant Feature Transform (SIFT) algorithm computes and detect the local features in images. Each new described feature from the new image is compared with already stored features in the database in order to know the common points between the two. A sparse point cloud is generated using the incremental SfM and is given to MVS

MVS on the other hand generates dense point cloud using the depth maps. It constitutes of densify point-cloud reconstruction for obtaining a complete and accurate as possible, point-cloud mesh reconstruction for estimating a mesh surface that explains the best the input point-cloud, mesh refinement for recovering all fine details, mesh texturing for computing a sharp and accurate texture to color the mesh.

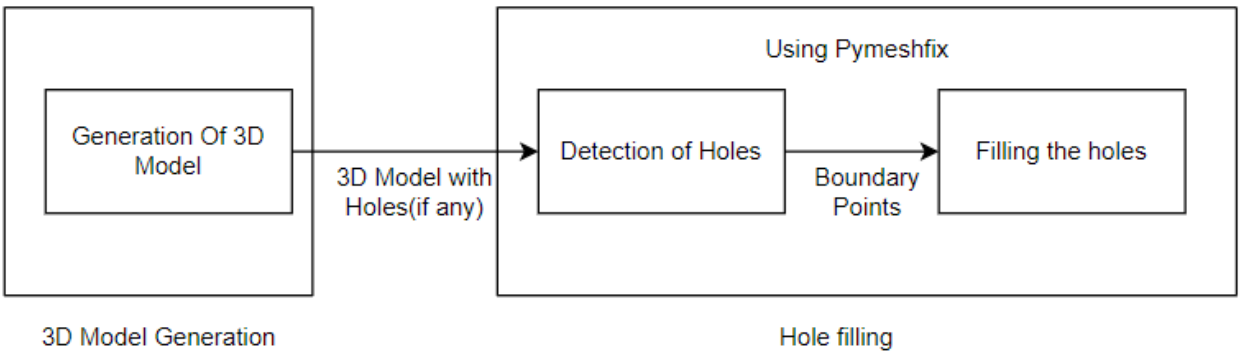

Figure 3.2: Removal of Holes

Figure 3.2 shows the the process of removal of holes. Holes generation and filling is done using pymeshfix. This open sourced software detects and fixes the holes in the model using complex algorithms which supports it. Each step is given in detail in the next implementation chapter.

Chapter 4

IMPLEMENTATION

This chapter gives a brief description of implementation details of the work by describing each module in detail. The main objective is to generate 3D models for a set of 2D input images with minimum holes. During the past few decades, image-based 3D reconstruction has advanced tremendously. As a result, the community gained access to a plethora of free and open-source solutions, with Photo Tourism being one of the pioneers in the sector. The objective is achieved by using OpenMVG and OpenMVS. OpenMVG and OpenMVS are open-source image-based 3D-reconstruction pipelines. Based on conventional multiple view geometry concepts, OpenMVG provides a full SfM pipeline. SIFT and AKAZE are used for feature detection and description. Classic brute force, ANN-kD trees, and cascade hashing are used to match features. Geometric verification of image pairs is implemented using homography, essential or fundamental matrix. Sparse reconstruction is computed using incremental or global techniques, and then bundle adjustment is done using the Ceres solver. The OpenMVS library implements dense reconstruction for large-scale scenes using a patch-based stereo approach for this pipeline combination.

The entire work was divided into two modules. The first is the generation of a 3D model for a set of 2D images and the second is the filling of holes. Further subsections include a detailed explanation of two modules.

4.1 Generation of 3D Model

3D reconstruction from multiple pictures is the process of creating three-dimensional models from a collection of images. The essence of a picture is a projection of a three-dimensional scene onto a two-dimensional plane, in which the depth is lost. The 3D point associated with a given image point must be on the line of sight. It is hard to tell which point on this line corresponds to the image point from a single picture. If two pictures are provided, the intersection of the two projection rays can be used to determine the position of a 3D point. Triangulation is the term for this procedure. The relationships between multiple views are crucial in this process because they provide the knowledge that corresponding sets of points must have some structure and that this structure is connected to the camera poses and calibration.

The 3D reconstruction is achieved by using structure from motion(SfM) followed by multi-

view stereo vision(MVS). Sfm is done using the OpenMVG library while MVS is achieved by the OpenMVS library. The input of this module is a set of 2D images of the monument and the output obtained will be a dense point cloud, that is, a 3D view of the input monument.

4.1.1 Structure from Motion

SfM algorithms take a collection of pictures as input and output each image's camera parameters and a set of three-dimensional points present in the images that are frequently stored as tracks. The 3D coordinates of a reconstructed 3D point and the list of matching 2D coordinates in a subset of the input pictures are defined as a track. The majority of today's SfM algorithms use the same fundamental processing pipeline.

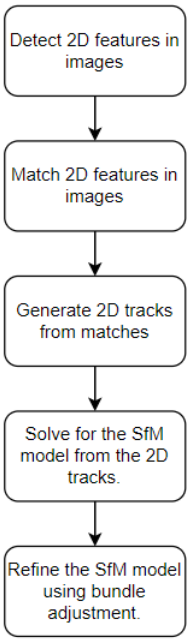

Figure 4.1: Main Stages of SfM Pipeline

Figure 4.1 represents a standard SfM pipeline's main phases. It starts with the detection of similar features and outputs the camera positions and orientation which are further used to generate a 3D model. SfM pipeline in OpenMVG follows a four-step process. The processes are image listing, image description computation, corresponding images, and correspondences computation, and SfM solving.

Image Listing

In OpenMVG pipelines, the first step in processing images is to create a sfmdata.json file that describes the image dataset that is used. The file created is a structured file. This well-organized file includes a View object for each image. The picture name, image size, and

internal camera calibration information, that is, the intrinsic parameters, are all stored and listed in this view. It will generate a sfmdata.json file, which openMVG will utilize as a scene description.

The required arguments to the image listing function are the path of the input image directory, the sensor width database text file, and the path of the output directory where the computed matches will be stored. The sensor width database is a text file that contains the details of the camera and its sensor width. JPEG EXIF metadata and a database of camera sensor width are used to perform intrinsic picture analysis. If the image has no metadata, we can either manually enter the known pixel focal length value or let the SfM algorithm discover it for us. If we choose the latter then at least two pictures must be defined that share common keypoints and belong to a valid intrinsic group. If the exchangeable image file format(EXIF) camera model and maker are identified in the provided sensor width database, the focal length in a pixel is calculated as follows:

$$focal_{pix} = \frac{max(w_{pix}, h_{pix}) * focal_{mm}}{ccdw_{mm}} \tag{4.1}$$

where:

$focal_{pix}$ = EXIF focal length in pixels
$focal_{mm}$ = EXIF focal length in mm
w_{pix}, h_{pix} = width and height of image in pixels respectively
$ccdw_{mm}$ = sensor width size in mm

We can compute the image featured once we have computed the dataset description using the next step in SfM.

Image Description Computation

Once the sfm data.json file is generated in step 1, this step computes an image description. It computes the picture description, that is, local areas for each view, and saves it to disc. The required arguments for this function are the path of the sfmdata.json file and an output directory where the image descriptions will be stored.

The extraction of features is a crucial stage in the 3D reconstruction process. The fundamental strategy is to locate the most distinctive keypoints or characteristics on each image. Once the characteristics have been identified, they may be used to compare or match them in various pictures. Because of its rotation and scale invariance, we utilize the SIFT keypoint detector. SIFT features are extracted from a collection of reference pictures and stored in a database for image matching and identification. A fresh picture is matched by comparing each feature from the new image to the old database and discovering possible matching features based on their feature vectors' Euclidean distance.

Corresponding Images and Correspondences Computation

We construct the related putative photometric matches using picture descriptions calculated in the previous step and filter the resultant correspondences using a few effective geometric filters. This phase entails searching the two pictures chosen and locating the pixel points that are matched. Feature matching can be accomplished with a brute force approach, ANN-KD trees, cascade hashing, or any nearest neighbor matching method. We use HNSWL2 (Approximate Matching with Hierarchical Navigable Small World Graphs) as our nearest neighbor algorithm.

The Euclidean distances between the feature vectors are used to quantify the similarity of key points in distinct digital pictures once the SIFT feature vectors of the key points are generated. A feature point from one picture is picked, then the shortest and next-shortest Euclidean distances in another image are traversed to get another two feature points. If the divisor between the shortest and next-shortest Euclidean distances between these two feature points is less than a threshold value, they are considered paired homologous feature points. The calculated matches are stored and used in further steps.

SfM Solving

Incremental SfM is used in the reconstruction process. The process begins with a two-view reconstruction which is then iteratively expanded by adding fresh views and three-dimensional points by using pose estimation and triangulation. Because the process is incremental, repeated non-linear refining stages, such as Bundle Adjustment (BA) are used to reduce the error generated.

Algorithm 1 Incremental Structure from Motion

Require: internal camera calibration and pairwise geometry consistent point correspondences.

Ensure: 3D point cloud and camera poses.

```
compute correspondence tracks t
compute connectivity graph G (1 node per view, 1 edge when enough matches)
pick an edge e in G with sufficient baseline
robustly estimate essential matrix from images of e triangulate validated tracks, which provides an initial reconstruction contract edge e
while G contains an edge do
    pick edge e in G that maximizes union(track(e),3D points)
    robustly estimate pose (external orientation/resection)
    triangulate new tracks
    contract edge e
    perform bundle adjustment
end while
```

A sfmdata.json file and certain pre-computed matches are used to run the chain. The sfmdata.json file, the location to the directory where geometric matches are kept, and the path to the output directory where the output data will be placed are all necessary parameters for this method. The algorithm 1 shows the implementation of Incremental SfM.

4.1.2 Multi View Stereovision

OpenMVS library is used to execute MVS.MVS algorithms are used to compute dense representations of the scene, resulting point clouds, once OpenMVG has estimated the camera locations and orientation.OpenMVS lets us compute dense points cloud, surface, and textured surfaces.OpenMVS attempts to provide a comprehensive set of techniques for recovering the whole surface of the scene that has to be generated. The input is a collection of camera positions and a sparse point cloud, with a textured mesh as the output. The processes that this library comprises are:

- **Dense point-cloud reconstruction:**The method generates a complete and precise point cloud.
- **Mesh Reconstruction:**This method is used to compute a mesh surface that best describes the input point cloud.
- **Mesh Refinement:**This procedure is used to restore all of the tiny details.

- **Mesh Texturing:**This procedure is used to create a fine and accurate texture that will be used to colour the mesh.

A sparse point cloud is converted to a dense point cloud in four phases. Image pair selection is the initial stage, followed by depth map computation, depth map refining, and depth map merging. In the stereo pair selection step, for stereo computation, we must choose a reference image for each image in the image collection. The choice of stereo image pair is critical not only for stereo matching accuracy but also for the final MVS outcome. In the depth map computation step, We compute the depth map for each stereo pair that qualifies. The basic notion is that we aim to identify a suitable support plane for each pixel in the input image that has the lowest aggregated matching cost with the reference image. Because the raw depth maps may not agree on common regions owing to depth inaccuracies, a refining procedure is used to ensure uniformity among surrounding views. In the last step, all the depth maps are merged to represent the scene.

4.2 Hole Filling

In locations with poor texture information, where no matching points can be discovered, the SfM method fails to estimate 3D information. As a result, the most prominent 3D reconstruction approaches provide effective results only if the target object's surface has enough texture. Virtual holes develop in the predicted models if this is not done. Due to surface reflectance, occlusions, and accessibility restrictions, holes may emerge in the resultant models.

To fill the holes in the generated meshes, we utilize a mesh-based hole filling technique. The algorithm takes as input a mesh consisting of a vertex set and a triangle set. A triangular mesh is made up of several vertices and triangles. Usually, two triangles, known as adjacent triangles of the edge, share one edge. An edge that is exclusively adjacent to one triangle is defined as a boundary edge. The boundary edge loop, on the other hand, is a representation of a closed hole border that may be automatically retrieved in the input mesh after a boundary edge is discovered by tracing its neighboring edges.[10]

We use the pymeshfix module to fill holes. This module is a Python/Cython wrapper of Marco Attene's MeshFix software[11]. This module combines C++ with the portability and simplicity of Python installation. It is an open-source project that uses pyvista.[12]PyVista is a utility module for the Visualization Toolkit (VTK) that uses NumPy anddirect array access to interact with VTK uniquely. This program provides a Pythonic, well-documented interface for quickly prototyping, analyzing, and visualizing spatially referenceddatasets.

Chapter 5

RESULTS AND DISCUSSIONS

The IHDS image dataset has been used for the reconstruction of 3D model. Under generation of the 3D model module, the 3D models were generated using OpenMVG and OpenMVS pipelines. The holes detected in the generated models were later filled in hole filling module using pymeshfix package.

5.1 Dataset Details

The dataset used in the project is the IHDS dataset.It consists of images of various heritage sites from different parts of the country.The size of the dataset is around 300GB. It is a crowd sourced dataset. It consists about 50 classes of temples in India where each class has around 500 images.

Figure 5.1: Sample images of Ambigera Gudi, Aihole

Figure 5.2: Sample images of Doddabasappa Temple, Gadag

Figure 5.1 and 5.2 represent sample images of two classes from the dataset. Figure 5.1 shows the samples images from Ambigera Gudi, Aihole class whereas Figure 5.2 shows the sample images from Doddabasappa Temple, Gadag.

5.2 Results Obtained

This section shows the results obtained for a few classes of the dataset. The sample input images used, the 3D model generated and the 3D models after hole filling are shown below.

Ambigera Gudi, Aihole

Figure 5.3: Sample input images of Ambigera Gudi, Aihole

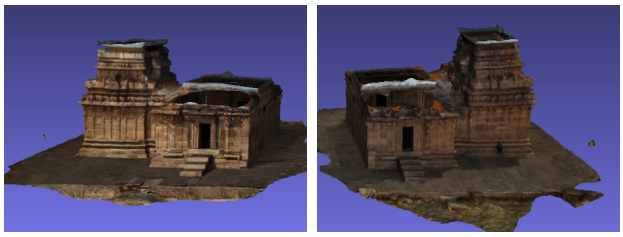

Figure 5.4: 3D model constructed for Ambigera Gudi, Aihole

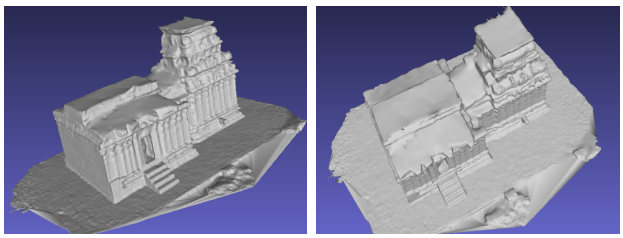

Figure 5.5: 3D model of Ambigera Gudi, Aihole after hole filling

Figure 5.3 shows the sample input images for generation of 3D model for Ambigera Gudi, Aihole. Figure 5.4 shows the 3D mode generated for the temple. It can observed that few holes are present in generated model.Figure 5.5 shows the 3D model after filling the holes.

Billeshwar Temple, Hanagal

Figure 5.6: sample images of Billeshwar Temple, Hanagal

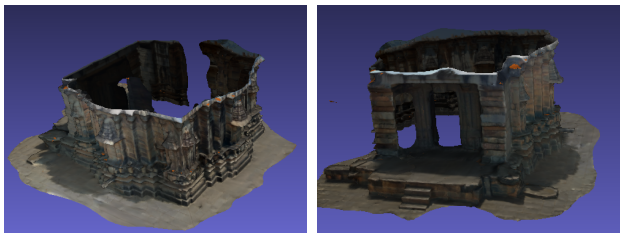

Figure 5.7: 3D model constructed for Billeshwar Temple, Hanagal

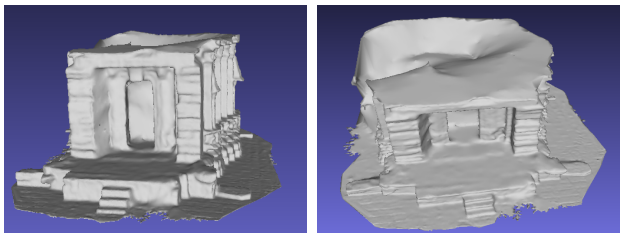

Figure 5.8: 3D model of Billeshwar Temple, Hanagal after hole filling

Figure 5.6 shows the sample input images for generation of 3D model for Keshava Temple, Mysore. Figure 5.7 shows the 3D mode generated for the temple. It can be observed that few holes are present in generated model.Figure 5.8 shows the 3D model after filling the holes.The holes in the model generated were generated as a result of nonavailability of images of those parts. It can be observed that due to the presence of such complex holes, hole filling is not really effective.Such holes can be filled by collecting images of missing parts of the temple.

Keshava Temple, Mysore

Figure 5.9: sample images of Keshava Temple, Mysore

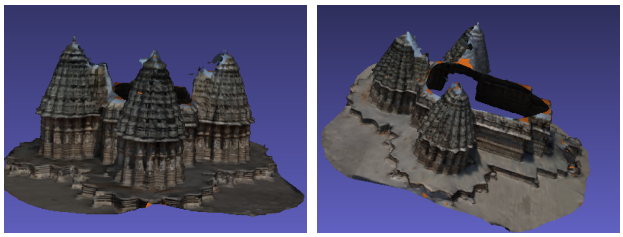

Figure 5.10: 3D model constructed for Keshava Temple, Mysore

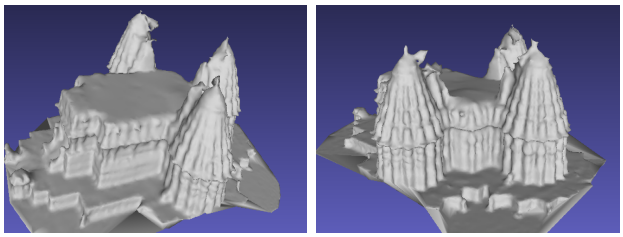

Figure 5.11: 3D model of Keshava Temple, Mysore after hole filling

Figure 5.9 shows the sample input images for generation of 3D model for Keshava Temple, Mysore. Figure 5.10 shows the 3D mode generated for the temple. It can observed that few holes are present in generated model.Figure 5.11 shows the 3D model after filling the holes.

Mahadev Temple, Tambdi Surla, Goa

Figure 5.12: Sample input images of Mahadev Temple, Tambdi Surla, Goa

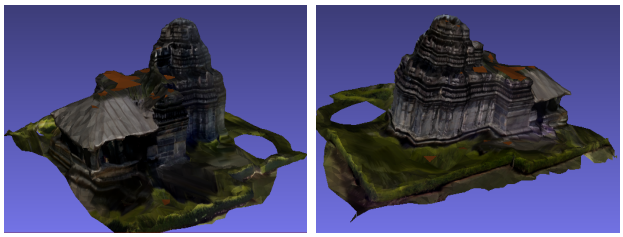

Figure 5.13: 3D model constructed for Mahadev Temple, Tambdi Surla, Goa

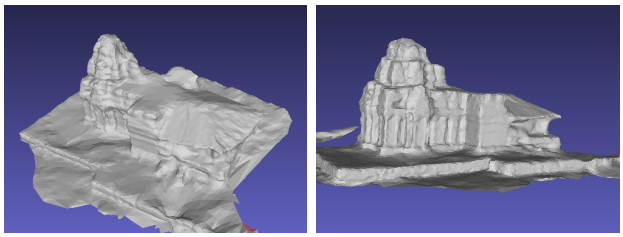

Figure 5.14: 3D model of Mahadev Temple, Tambdi Surla, Goa after hole filling

Figure 5.12 shows the sample input images for generation of 3D model for Mahadev Temple, Tambdi Surla, Goa. Figure 5.13 shows the 3D mode generated for the temple. It can observed that few holes are present in generated model.Figure 5.14 shows the 3D model after filling the holes.

Chapter 6

CONCLUSION AND FUTURE SCOPE OF THE WORK

The 3D models for different architectural heritage sites in the IHDS dataset were generated using OpenMVG and OpenMVS open-source image-based 3D reconstruction pipelines. Holes were found in the 3D models obtained due to featureless surfaces and missing information of that particular area. The holes obtained were filled using pymeshfix, a python package to repair meshes.

However, still some issues can be identified in the proposed method.The performance can be further improved by collecting more images for a heritage site in the dataset.Collection of more images from all angles and capturing all the features might lead to reduction of holes obtained in the 3D models.

REFERENCES

[1] Klaus Häming and Gabriele Peters. The structure-from-motion reconstruction pipeline–a survey with focus on short image sequences. *Kybernetika*, 46(5):926–937, 2010.

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

[3] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[4] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.

[5] Shuhan Shen. Accurate multiple view 3d reconstruction using patch-based stereo for large-scale scenes. *IEEE transactions on image processing*, 22(5):1901–1914, 2013.

[6] Shawn McCann. 3d reconstruction from multiple images. *Stanford Computational Vision and Geometric Lab*, 2015.

[7] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM siggraph 2006 papers*, pages 835–846. 2006.

[8] Elisavet Konstantina Stathopoulou and Fabio Remondino. Open-source image-based 3d reconstruction pipelines: Review, comparison and evaluation. In *6th International Workshop LowCost 3D–Sensors, Algorithms, Applications*, pages 331–338, 2019.

[9] Emiliano Pérez, Santiago Salamanca, Pilar Merchán, and Antonio Adán. A comparison of hole-filling methods in 3d. *International Journal of Applied Mathematics and Computer Science*, 26(4), 2016.

[10] Ying Wang Xiaoyuan Guo, Jun Xiao. A survey on algorithms of hole filling in 3d surface reconstruction. *The Visual Computer*, 34:93–103, 2018.

[11] Marco Attene. A lightweight approach to repairing digitized polygon meshes. *The Visual Computer*, 26(11):1393–1406, 2010.

[12] C. Bane Sullivan and Alexander Kaszynski. PyVista: 3d plotting and mesh analysis through a streamlined interface for the visualization toolkit (VTK). *Journal of Open Source Software*, 4(37):1450, may 2019.

Appendix A

A.1 Fundamental Matrix

The basic matrix is a connection between any two pictures of the same scene that limits the location of point projections from the scene in both images. The matching point in the other picture is limited to a line when a scene point is projected into one of the images, aiding the search and enabling for the identification of incorrect correspondences. Discrete matching constraint, matching constraint, Epipolar constraint, or incidence relation refers to the relationship between comparable image points that the fundamental matrix depicts. A set of point correspondences can be used to derive the fundamental matrix. Additionally, using camera matrices generated directly from this fundamental matrix, these related picture points may be triangulated to world locations. The scene made up of these world points is a projective modification of the original scenario.

A.2 SIFT Detector

Once the keypoint is found, the next step is to construct a descriptor that contains information of visual characteristics around the keypoint yet is not sensitive to rotation and image illumination. The steps of building the SIFT descriptor are as following:

- Use the Gaussian blurred image associated with the key point's scale.
- Take image gradients over a 16x16 array.
- Rotate the gradient directions and locations relative to the keypoint orientation.
- Create an array of orientation histograms.
- Add the rotated gradients into their local orientation histograms with 8 orientation bins.
- The resulting SIFT descriptor is a length 128 vector representing a 4x4 histogram array with 8 orientation bins per histogram.

Rotation and scale have no effect on the SIFT descriptor. The vector's robustness to variations in illumination can be increased by normalising and clamping it. Using Euclidean

distance between descriptor vectors, the SIFT vectors may be used to compare key points from image A to key points from image B to discover matching keypoints.

Because of its invariance to picture changes, we utilise the SIFT keypoint detector. Several thousand SIFT keypoints may be found in a single picture. Then, using approximate closest neighbours module, we match keypoint descriptions between each pair of pictures. Then, using RANSAC, reliably estimate a basic matrix for the pair. Using the eight-point technique, we compute a candidate fundamental matrix during each RANSAC iteration, followed by non-linear refinement. Finally, matches that are outliers to the reconstructed basic matrix are removed. We exclude all matches from consideration if the number of remaining matches is fewer than twenty.

We arrange the matches into tracks after identifying a set of geometrically consistent matches between each picture pair. A track is a linked group of matching keypoints across several photos. A track is termed inconsistent if it has more than one keypoint in the same picture. For the following part of the reconstruction method, we preserve consistent tracks with at least two keypoints.