AliceVision Meshroom: An open-source 3D reconstruction pipeline

Carsten Griwodz, Simone Gasparini, Lilian Calvet, Pierre Gurdjos, Fabien Castan, Benoit Maujean, Yann Lanthony, Gregoire de Lillo

AliceVision Meshroom: An open-source 3D reconstruction pipeline

Carsten Griwodz
griff@uio.no
University of Oslo
Norway
SimulaMet
Norway

Simone Gasparini
Lilian Calvet
Pierre Gurdjos
{name.surname}@irit.fr
University of Toulouse, IRIT
France

Fabien Castan
Benoit Maujean
Yann Lanthony
Gregoire De Lillo
{name.surname}@mikrosimage.com
Mikros, Technicolor Creative Studios
France

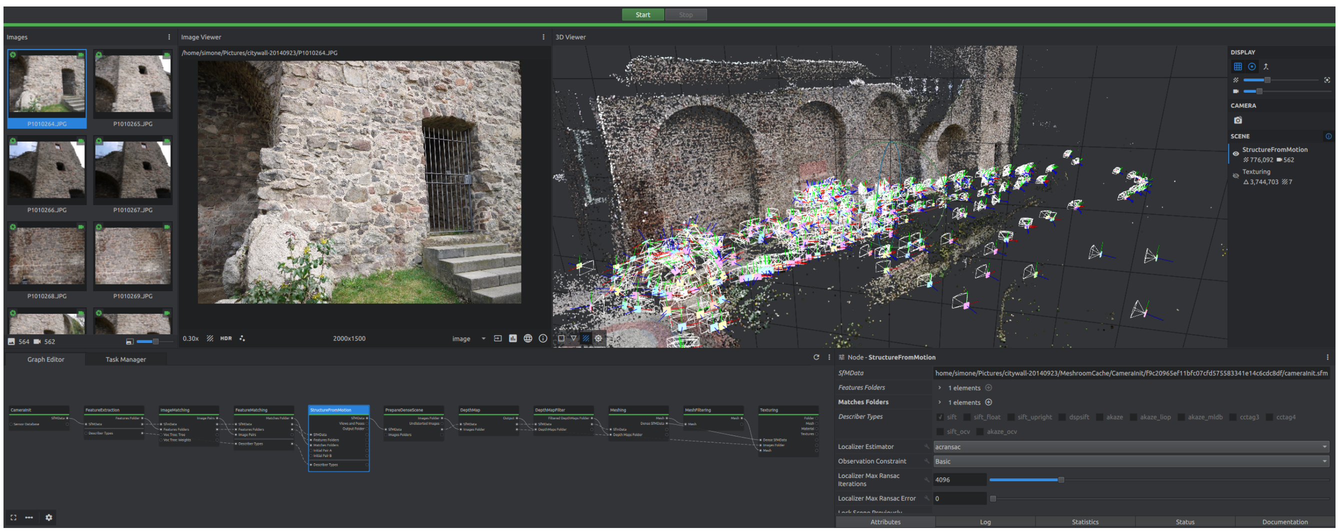


Figure 1: A 3D reconstruction pipeline in Meshroom. The pipeline is shown at the bottom left, the input images on the top left, the output of the highlighted pipeline node (in this case the Structure-from-Motion node along with the camera poses) are shown on the top right, info about this node at the bottom right.

ABSTRACT

This paper introduces the Meshroom software and its underlying 3D computer vision framework AliceVision. This solution provides a photogrammetry pipeline to reconstruct 3D scenes from a set of unordered images. It also features other pipelines for fusing multi-bracketing low dynamic range images into high dynamic range, stitching multiple images into a panorama and estimating the motion of a moving camera. Meshroom's nodal architecture allows the user to customize the different pipelines to adjust them to their domain specific needs. The user can interactively add other processing nodes to modify a pipeline, export intermediate data to analyze the result of the algorithms and easily compare the outputs given by different sets of parameters. The software package is released in open source and relies on open file formats. These features enable researchers to conveniently run the pipelines, access and visualize the data at each step, thus promoting the sharing and the reproducibility of the results.

CCS CONCEPTS

• **Information systems** → *Multimedia content creation*; • **Computing methodologies** → **Reconstruction**.

KEYWORDS

open source, 3D computer vision, photogrammetry, nodal framework, camera tracking, HDR imaging, panorama, point clouds

ACM Reference Format:

Carsten Griwodz, Simone Gasparini, Lilian Calvet, Pierre Gurdjos, Fabien Castan, Benoit Maujean, Yann Lanthony, and Gregoire De Lillo. 2021. AliceVision Meshroom: An open-source 3D reconstruction pipeline. In *MMSys '21: ACM MUltimedia Systems Conference, Sept 28–Oct 01, 2021, Istanbul, Turkey.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3458305.3478443

MMSys '21, Sept 28–Oct 01, 2021, Istanbul, Turkey

ACM ISBN 978-1-4503-8434-6/21/09…$15.00
https://doi.org/10.1145/3458305.3478443

1 INTRODUCTION

Multimedia systems research is increasingly dealing with 3-dimensional content in its many facets. Researchers' interests include such widely different applications as the streaming of 360-degree video that allows free movement of the viewer in 3 rotational and 3 translational dimensions (6DOF) [1, 2] and adaptive point cloud streaming [3]. They study issues of latency during the interaction in virtual worlds [4] or the compression of meshes [5]. Due to the wide range of interests and the difficulty in creating content, there is little exchange in terms of content and tools that could help more researchers to join these investigations.

3D Computer Vision is used in many industries and research domains for various use cases with very different acquisition setups. It is challenging to address all the needs in a single solution, but there is also a great convergence of the needs regarding the low level building blocks that can be used for these different pipelines. We are therefore proposing the Meshroom[1] software with its underlying 3D Computer Vision framework, AliceVision[2]. It provides a photogrammetry pipeline to reconstruct 3D scenes from still images taken with any type of cameras, from professional cameras to smartphones and can deal with an arbitrary number of input images, ranging from a handful to several thousands. Meshroom also provides other pipelines, for example for fusing multi-bracketing low dynamic range (LDR) images into high dynamic range (HDR) images, stitching multiple images into a panorama and estimating the motion of a moving camera. To fulfil such a variety of tasks, Meshroom is designed around a nodal engine. This enables end-users to customize the pipeline to their specific needs, even allowing them to extend it by adding their own nodes. The extensibility is supported by Meshroom's reliance of standardized and open-source formats, which has helped to integrate other tools into the Meshroom pipeline.

Meshroom and AliceVision are released in open source under the Mozilla Public License v2 (MPLv2). The license does not limit the use of either the software itself or its products. Meshroom can be used to generate content for a wide variety of multimedia research topics and hopefully share it with the community. With its modular architecture, it can also be extended with domain specific features by the different communities using dedicated nodes.

In this paper, we refer to release 2021.1.0 of Meshroom. It can be found on Github at **https://git.io/JZkxK**. The project is under active development and newer releases can be expected. The manual for Meshroom can be found on readthedocs.io at https://meshroom-manual.readthedocs.io. On YouTube, you can find videos that explain how to use Meshroom for 3D reconstruction at https://youtu.be/1U0g_zxVDdg and for HDR 360 panorama creation at https://youtu.be/WLrB1eiw3Cc. Meshroom is developed in Python and it uses Qt/PySide for the graphical interface. The AliceVision framework is developed in C++ and the depth map tool is implemented in C++/Cuda[3]. Meshroom can run on both Linux and Windows[4].

[1] http://meshroom.alicevision.org
[2] https://alicevision.org
[3] The latest release is compiled for CUDA SDK 10.2.
[4] It can also run on MacOS, but this is currently not officially supported because of the lack of CUDA support on MacOS in recent releases.

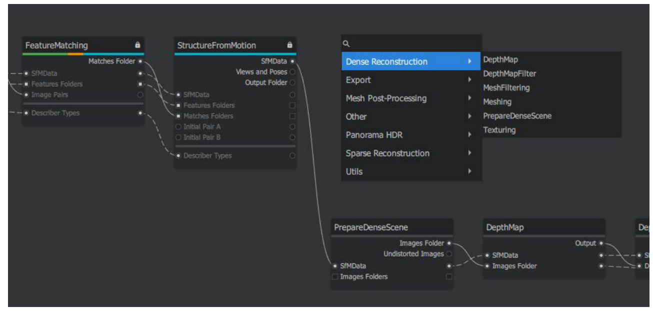


Figure 2: In the node editor the user can edit the pipeline and add new nodes

2 BACKGROUND

The history of AliceVision lies with a series of research projects and teams that joined forces. AliceVision's SfM stage has its origin in a collaboration of IMAGINE (a joint research group between Ecole des Ponts ParisTech and Centre Scientifique et Technique du Bâtiment) and the post-production company Mikros Image that led to the creation of the open-source project OpenMVG [6]. The MVS stage has its origin in the closed-source tool CMPMVS that was created at the Czech Technical University in Prague (CTU) [7]. The accurate marker-based tracking system CCTag [8] was created in a collaboration of the University of Toulouse (INP), INRIA and the company DuranDuboi. Simula implemented a real-time GPU implementations of CCTag and SIFT (PopSift [9]) in the EU project #644874 POPART.

Meshroom has been used since 2014 in digital environment creation for the Visual Effects industry and now in many other industries including manufacturing, medical [10], cultural heritage [11], tourism , archaeology [12, 13], biology [14], surveillance [15] and 3D printing.

Several other open source solutions for 3D reconstruction with a GUI are available. To the best of our knowledge we can mention COLMAP [16], MicMac [17], MVE [18], OpenDroneMap [19], Regard3D [20] (based on openMVG and openMVS).

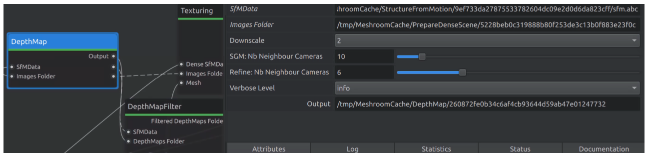
Many commercial solutions exist for 3D reconstruction: **ArcGIS** by ESRI, **ContextCapture** by Bentley Systems, **Correlator3d** by Simactive, **Inpho** by Trimble, **iWitnessPRO** by Photometrix, **Metashape** by Agisoft, **Pix4DMapper** by Pix4D, **PFTrack** by ThePixelFarm, **RealityCapture** by Epic Games, **ReCap** by Autodesk and **Zephyr** by 3DFlow.

AliceVision has been conceived as a complete open source photogrammetry framework that bridges the gap between academic research and professional demands. To ensure long-term governance of AliceVision as an open source project, the AliceVision Association[5] was founded in 2020.

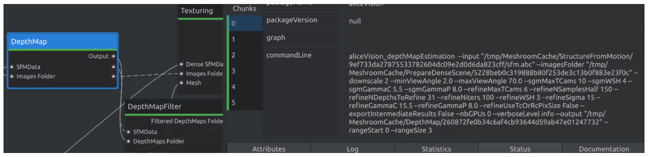
3 MESHROOM SOFTWARE

Meshroom provides a nodal environment to perform various computer vision tasks. The individual tasks are represented by nodes combined into directed acyclic dependency graphs that are named pipelines. Each node represents a tool implemented in AliceVision.

[5] https://alicevision.org/association



(a) Attributes for DepthMap node



(b) Command line string for DepthMap node

Figure 3: Example configuration for configuring a node

Unlike pipelines that are meant for real-time processing of data streams, the nodes in Meshroom are logical steps in the directed graph that are revisited when inputs or parameters change or when the graph itself is modified. The changes invalidate intermediate results cached in nodes only on the downstream side of the node where the change happened. Each node may represent an operation on hundreds or thousands of input files, and hide parallel processing operations. Meshroom is designed to optionally run the pipeline on a computer cluster and is for instance used with the commercial Tractor render farm system by Pixar. The multi-machine parallelization can be done between nodes but also the work of a single node can be split into multiple parallelizable sub-tasks. Final and intermediate file formats rely on popular open formats such as OBJ, EXR and Alembic to facilitate interoperability. Multiple plugins have been contributed for the integration in production tools, such as Maya (Autodesk), Houdini (SideFX) and Blender (Blender Foundation). The importance of the nodal architecture for Meshroom is reflected in the GUI, where the node editor (Figure 2) allows users to monitor progress, inspect parameters, visualize outputs and change the pipeline interactively.

While Meshroom provides preconfigured pipelines, users can change it arbitrarily and store their configuration as the new default pipeline. It is possible to add or remove nodes in the pipeline, or to connect several downstream nodes to the output of a stage. The latter allows, for example, to experiment with the settings of a critical node in the pipeline by executing it with several parameter sets within a single workflow. The workflow can furthermore be interrupted and combined with exporter and importer nodes. This enables the manual inspection of intermediate results or the modification with specialized external tools by the end-user. Steps where this is frequently exploited by graphic artists so far are the later stages of the pipeline (mesh post-processing steps and texturing), but it is possible in arbitrary steps and pipeline configurations. Users can inspect the parameters of each tool in the pipeline and modify them interactively, or retrieve the string to run the individual task from the command line for even more detailed control. Examples for this are shown in Figures 3a and 3b, respectively.

Meshroom pipelines can also be computed from the command line without the graphical interface, which makes it completely scriptable. For instance, the end-user can run the following command line:

```
> ./meshroom_batch \
                --input    ./my_dataset \
                --pipeline ./my_custom_pipeline.mg \
                --save     ./scene/my_scene.mg \
                --output   ./my_folder
```

4 PHOTOGRAMMETRY PIPELINE

The main purpose of AliceVision is to allow photogrammetric 3D reconstruction from an arbitrary number of images. The two main stages of its default pipeline are "Structure-from-Motion" (SfM) and "Multi View Stereo" (MVS) although it is also possible to create textured meshes directly from the SfM stage. These stages are composed of a variety of nodes. It comprises the steps of camera initialization, feature extraction, image matching, feature matching, Structure-from-Motion, depth mapping (consisting of preparation, mapping and filtering), meshing followed by mesh filtering, and texturing.

4.1 Feature extraction

Feature extraction forms the basis for finding the relative pose of cameras in space by detecting distinctive points in every image that can be described by properties that are invariant to scale, rotation in 2D and 3D and several other properties such as lighting. AliceVision supports both natural feature extractors (SIFT [22], DSP-SIFT [23] and AKAZE [24]) and marker-based features (CCTag [25], AprilTag [26]), and can use a combination of them in later stages to benefit from their different properties. Markers can help the reconstruction process in textureless environments by providing a reliable detection accuracy and improving the matching process. They can be also used to bring the reconstructed 3D model to the metric scale if the relative distances of the markers are known (see Section 5.1).

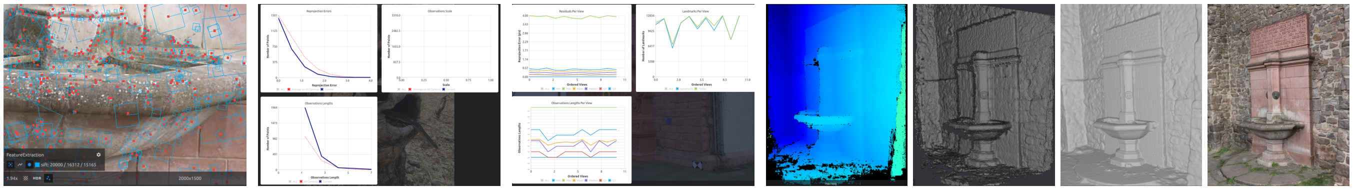
The user can display the extracted features, possibly for each type if multiple types of features were extracted, along with their photometric properties like the scale and the orientation. Features that were extracted but not used in the reconstruction process are identified with a different color and the user can select which visualization to display and which type(s) of feature to display (Figure 4a).

4.2 Image matching

In principle, to find corresponding points among the images, the features of each pair of images must be compared and matched. As the number of input images increases such comparison does not scale due to the quadratic complexity of the task, In order to reduce the computational complexity for large datasets, a vocabulary tree (VocTree) [27] is used to find subsets of images sharing some similar content and limit the matching only to those images. By default, AliceVision uses VocTrees when the number of images is larger than 200 images.

4.3 Feature matching

In this step, features are matched between pairs of images, which will subsequently allow to determine a 3D structure from the corresponding 2D positions. This step is trivial for marker based features



(a) (b) (c) (d) (e) (f) (g)

Figure 4: Meshroom provides several tools to help verify and understand the intermediate results, such as visualizing the extracted features (a), reconstruction statistics (b,c), the depth maps as images (d) or back-projected in 3D (e), the final mesh mesh with (g) and without the texture (f). Fountain dataset [21].

where only their unique IDs must be matched. For natural features, the task is more computationally expensive, due to the large number of features typically extracted from an image.

First, a matching between the feature descriptors is performed. For each descriptor in one image, we select the closest descriptor in the other image. As each descriptor should match with a single point in the other image, repetitive structures are discarded at this step. AliceVision implements multiple methods: brute force matching, Cascade Hashing [28] and a KD-tree-based approach using FLANN [29] used by default. Other approaches exist in the litterature, like Locality-Sensitive Hashing [30]. The cross-matching option performs the matching from the first image to the second and vice-versa, and keeps the match if it gives the same result. This is more strict and requires more computation time but improves the inliers/outliers ratio. For each pair of images, the descriptor matches are validated and filtered by checking whether the 2D location of the corresponding feature points are geometrically consistent regarding the epipolar geometry in a robust RANSAC [31] framework.

A guided-matching option allows recomputing the descriptor matches using the estimated relative poses. Also if a prior on the camera poses is provided, the guided matching can be used directly to help matching repetitive structures.

4.4 Structure from Motion

In the following Structure from Motion (SfM) step, the computed matches are fused together into tracks: each track is a candidate to represent a point in space, visible from multiple cameras. The tracks are used to solve the camera calibration and the 3D structure of the scene, thus generating a sparse 3D representation. The SfM stage can be addressed in several ways. The positions can be computed by incrementally adding new cameras from an initial solution [32], solving for all the relative poses of the cameras at once in a global solution [33], or taking a hierarchical [34, 35] or multi-stage [36] approach. AliceVision implements an incremental approach [6, 16] that is better suited for large numbers of images than the other approaches. AliceVision does also support camera rigs that consist of several cameras that are mounted in fixed relative positions to each other to add additional constraints to the pose estimation.

At the end of the SfM computation the statistics for the reprojection error, the number of reconstructed 3d points and the number of tracks are shown for each view (Figure 4c) or globally, for the entire reconstruction (Figure 4b).

4.5 Depth map estimation

This step attempts to estimate a depth value for each input pixel. The region needs to be seen by at least 2 cameras that have been validated by the SfM. AliceVision implements a Semi-Global Matching (SGM) [37] approach. The classical alternative methods are block matching and ADCensus [38]. The SGM method is a volumetric approach that estimates a global similarity score (ZNCC) over multiple images for each cell of a voxel grid. An optimization of the score volume is performed using cost aggregation to retrieve a first low resolution version of the depth map. Then the result is upscaled and refined with a brute force method. Finally a noise filtering is performed to align depth discontinuities with color variations. To increase the scalability of the depth mapping stage, AliceVision relies on GPU-based computation that depends on the NVidia CUDA SDK.

The depth maps estimated for each image can be visualized with a colormap for the depth both in the 2D viewer (Figure 4b) and in 3D viewer (Figure 4c) to visually assess the quality of the depth map projected onto the reconstructed scene.

4.6 Meshing

To create a dense 3D surface, the Meshing step will merge all depth maps into a single dense point cloud and then extract a surface in it. For each input image, we select the most interesting depth values (repartition in the image and good similarity score) and back-project them as 3D points. We use a KDTree approach [39] to fuse the 3D points together in an iterative way to reduce the density of the point cloud to fit in RAM.

3D Delaunay triangulation [40] is used to create a space filled by tetrahedra. Then a voting strategy is used to label each tetrahedron as full or empty. The weights of the tetrahedra and their connections are computed according to the work of Jancosek et al. [41, 42], then a graph cut max-flow step [43] is applied to extract the mesh surface. Local mesh artifacts are filtered and the surface is finally smoothed with a bilateral filtering.

4.7 Texturing

The final step of the 3D reconstruction pipeline projects the textures onto the mesh. AliceVision uses a basic UV mapping approach [44]. To select the cameras that provide the best texture of a particular cell of the mesh, AliceVision computes the resolution of the triangle in all images seeing it. The texture fusion uses a multi-band blending [45] using more images in the low frequencies than in

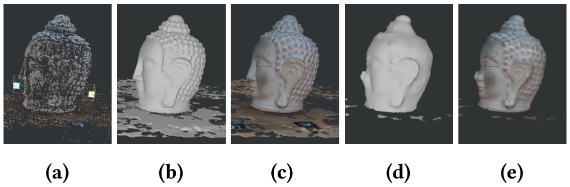


Figure 5: Bypassing dense mapping saves time and works without CUDA, at the price of quality. Example from 18 pictures of [48]. (a) SfM output; (b) Mesh after dense map; (c) Textured mesh after dense map; (d) Mesh when bypassing dense map; (e) Textured mesh when bypassing dense map.

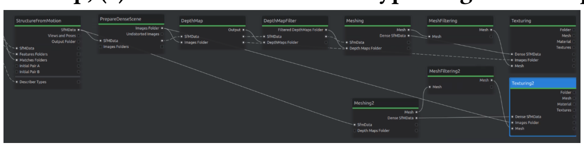


Figure 6: Pipeline with and without GPU bypass, Texturing and Texturing2 producing the results shown in Figure 5c and Figure 5e, respectively.

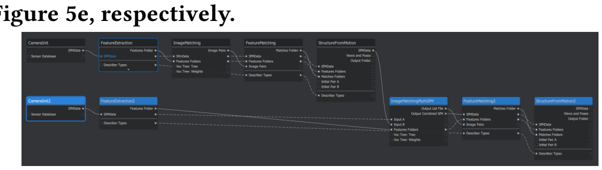


Figure 7: New images can be added through the augmentation pipeline (blue nodes).

the high frequencies to ensure the global color coherence while keeping the pixel precision of the fine texture details, in the same spirit as [46, 47].

5 PIPELINE CUSTOMIZATIONS

The variety of use cases results in a wide range of capturing devices, settings and environment constraints. This can include anything from a professional DSLR to a smartphone camera, from a single camera moving freely in a static scene to multiple synced cameras, mixed with projected light patterns, infrared light, light polarisation or controlled lighting variations, shooting in indoor or outdoor conditions, etc. This section describes several common customizations of the 3D reconstruction pipeline.

5.1 Scale to metric values

The result of the photogrammetry is metric up to a scale factor that cannot be solved without additional information in the scene. Using multiple markers (like CCtags) with a known distance in the scene, we can redefine the scale of the Structure-from-Motion result to a real-world metric. The same markers can also be used to define the orientation of the scene.

5.2 Light pattern projection

One of the major weaknesses of photogrammetry is the inability to reconstruct uniform surfaces. One solution is therefore to use a synchronized multi-camera system and take two shots with and without a projected light pattern (see Figure 8). The geometry can be estimated from the images with the projected pattern and the texturing can be done from the other set of images without any light pattern. If we reconstruct an object that is not static like a person, it is important that the 2 shots are fast enough to ensure that the person has not moved between the 2 acquisitions.

5.3 Quick rough Reconstruction

Many parameters could be adjusted to speedup the process at the cost of quality and robustness. As the most compute intensive step is the depth map estimation, we can get a quick preview of the mesh by skipping the depth maps computation and directly connecting the SfM to the Meshing node. Figure 5 illustrates the difference. A configuration with and without bypass is shown in Figure 6.

5.4 SfM Augmentation

Given a computed SfM node it is possible to augment the reconstruction by adding new images without recomputing the previous steps. Figure 7 shows that the output of the node can be used as input of another reconstruction pipeline as base to add the new images to further complete the model with new parts or improve some details.

5.5 Live reconstruction

Instead of processing all the images as a batch, it is possible to "watch" a given folder and perform the 3D reconstruction on the fly as soon as new images are transferred from the camera into the watched folder - using the SfM augmentation. This allows the user to view the reconstruction during the acquisition, so he can ensure that an object is fully covered without missing parts and avoid uneven camera coverage.

5.6 Camera tracking

The camera pose estimation provided by the SfM pipeline can be used for camera tracking. A photogrammetry of the location can be used as a reference structure. As described in Section 5.4, the Live Action camera images can be added to the images of the shooting set location. A dedicated pipeline using the SfM nodes is currently in development.

The result of the SfM can be exported as an animated camera in an Alembic file plus the corresponding undistorted images (using the `ExportAnimatedCamera` node).

6 HDR 360° PANORAMA PIPELINE

A classical example of HDR 360° images usage is the lighting of virtual scenes with Image Based lighting techniques (Figure 9). These 360° images cannot be captured directly and are the result of the assembly of different shots of the scene. The process of creating an HDR 360° panorama is done in three steps: (i) the HDR fusion by assembling, for each angle of view, the images taken at different exposure time values - called "multi-bracketing", (ii) the estimation of the internal and external parameters of the cameras and (iii) the stitching by assembling the different resulting HDR images into a panorama covering 360 degrees along the horizon line.

Figure 8: (a) Geometry and texturing estimated without (b) and with light pattern projection; (c) Geometry estimated with light pattern projection and texturing estimated without light pattern (dataset from [49]).

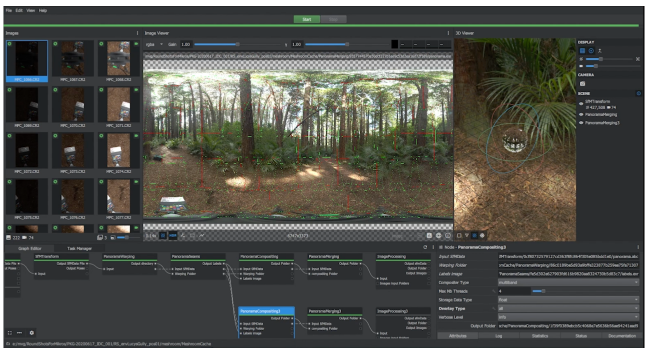


Figure 9: Example of the HDR 360° panorama pipeline.

6.1 LDR to HDR fusion

To fuse the LDR images into HDR images, we first select reliable pixels[6] and then calibrate the Camera Response Function (CRF) with 2 state-of-the-art methods: [50] and [51]. The merge step applies the CRF and fuses the LDR colors using a weighting function to favor stable values from the camera sensor.

6.2 Panorama estimation

In the context of the panorama, the camera motion is a pure rotation around the camera center. Similarly to the photogrammetry pipeline, feature points are extracted for each image and then matched to estimate the rotation matrices relating to the images. In a final optimization step, the internal parameters of the cameras and the rotations are refined with a Bundle Adjustment. Meshroom can also take advantage of motorized-head files to initialize camera poses before the final optimization.

6.3 Panorama stitching

The panorama can be created from fisheye optics. In that case, the image contains useless black pixels that should not become part of the final panorama. So first, it automatically estimates the fisheye circle (which can also be adjusted interactively) to select the useful part of the input images. The stitching implements a graph-cut algorithm [52] to optimize seams location. Then to create seamless transitions, the fusion is done with a multi-band blending [53] to ensure the color consistency between images (low-frequencies) while keeping all the details (high-frequencies). The stitching implements a tiling strategy to generate large panoramas like 70K resolution.

[6]Instead of selecting the pixels in a regular way on the whole image (1 pixel out of N), this step keeps the pixels carrying the most valuable information: low noise, increasing values with the exposure, no saturated value.

7 SAMPLE RESULTS

Table 1 presents some sample results from a selection of datasets of different sizes taken from the literature. For each dataset it shows the number of images, the processing time and the number of vertices and faces of the final model. The results are obtained on a single machine sporting an Intel® Core™ i9-10900K CPU @ 3.70 GHz with 20 cores, 32 GB RAM and equipped with a GeForce RTX 3080 with 10 GB of memory. The relevant 3D models can be found at the following link https://skfb.ly/o6LLp.

dataset	images	time	vertices / faces
Fountain [21]	10	2m 19s	242517 / 484383
Buddha [48]	67	28m 25s	15557379 / 3109710
Citywall [18]	564	1h 50m 20s	1875571 / 3744703

Table 1: Results obtained with the datasets.

8 CONCLUSION

The technologies for 3D digitization are becoming increasingly essential for many industries and are a key-enabler in many applications, including the commons like healthcare, cultural preservation and creation. This paper introduced a free 3D computer vision software, that adheres to open standards and can be freely used, analyzed and modified by everyone. This project ambition is to foster interactions between research and industry, as well as with education. There is room for researchers, developers and 3d scanning experts to implement complementary features such as georeferencing, multi-focus or depth sensors integration. A lot of ambitious research remains to be done to process moving objects, to reconstruct the lighting of the scene and recover the material properties of the surfaces.

REFERENCES

[1] Jong-Beom Jeong, Soonbin Lee, Il-Woong Ryu, Tuan Thanh Le, and Eun-Seok Ryu. Towards Viewport-dependent 6DoF 360 Video Tiled Streaming for Virtual Reality Systems. In *Proc. 28th ACM Int. Conf. Multimed.*, pages 3687–3695, New York, NY, USA, oct 2020. ACM.

[2] Savino Dambra, Giuseppe Samela, Lucile Sassatelli, Romaric Pighetti, Ramon Aparicio-Pardo, and Anne-Marie Pinna-Déry. Film editing. In *Proc. 9th ACM Multimed. Syst. Conf.*, pages 27–39, New York, NY, USA, jun 2018. ACM.

[3] Mohammad Hosseini and Christian Timmerer. Dynamic Adaptive Point Cloud Streaming. In *Proc. 23rd Pack. Video Work.*, pages 25–30, New York, NY, USA, jun 2018. ACM.

[4] Suraj Raghuraman, Kanchan Bahirat, and Balakrishnan Prabhakaran. A Visual Latency Estimator for 3D Tele-Immersion. In *Proc. 8th ACM Multimed. Syst. Conf.*, pages 272–283, New York, NY, USA, jun 2017. ACM.

[5] Cédric Portaneri, Pierre Alliez, Michael Hemmer, Lukas Birklein, and Elmar Schoemer. Cost-driven framework for progressive compression of textured meshes. In *Proc. 10th ACM Multimed. Syst. Conf.*, pages 175–188, New York, NY, USA, jun 2019. ACM.

[6] Pierre Moulon. *Positionnement robuste et précis de réseaux d'images.* PhD thesis, Laboratoire d'Informatique Gaspard-Monge, École des Ponts ParisTech, 2014.

[7] Michal Jančošek. *Large Scale Surface Reconstruction based on Point Visibility.* PhD thesis, Czech Technical University in Prague, 2015.

[8] Lilian Calvet. *Méthodes de reconstruction tridimensionnelle intégrant des points cycliques : application au suivi d'une caméra.* PhD thesis, Institut National Polytechnique de Toulouse, 2014.

[9] Carsten Griwodz, Lilian Calvet, and Pål Halvorsen. Popsift. In *Proc. 9th ACM Multimed. Syst. Conf.*, pages 415–420, New York, NY, USA, jun 2018. ACM.

[10] Toby Collins, Daniel Pizarro, Simone Gasparini, Nicolas Bourdel, Pauline Chauvet, Michel Canis, Lilian Calvet, and Adrien Bartoli. Augmented Reality Guided Laparoscopic Surgery of the Uterus. *IEEE Transactions on Medical Imaging*, 40(1):371–380, January 2021.

[11] Asla Medeiros e Sá, Adolfo Bartolome Ibañez Vila, Karina Rodriguez Echavarria, Ricardo Marroquim, and Vivian Luiz Fonseca. Accessible Digitisation and Visualisation of Open Cultural Heritage Assets. In Selma Rizvic and Karina Rodriguez Echavarria, editors, *Eurographics Workshop on Graphics and Cultural Heritage.* The Eurographics Association, 2019.

[12] J. Milàn, P. Falkingham, and Inken Juliane Mueller-Töwe. Small ornithopod dinosaur tracks and crocodilian remains from the middle jurassic bagå formation, bornholm, denmark: Important additions to the rare middle jurassic vertebrate faunas of northern europe. *Bulletin of The Geological Society of Denmark*, 68:245–253, 2020.

[13] Jens Lallensack, Michael Buchwitz, and Anthony Romilio. Photogrammetry in ichnology: 3d model generation, visualisation, and data extraction. November 2020.

[14] Shah Ariful Hoque Chowdhury, Chuong Nguyen, Hengjia Li, and Richard Hartley. Fixed-lens camera setup and calibrated image registration for multifocus multiview 3d reconstruction. *Neural Computing and Applications*, April 2021.

[15] Marco Wallner, Daniel Steininger, Verena Widhalm, Matthias Schörghuber, and Csaba Beleznai. RGB-d railway platform monitoring and scene understanding for enhanced passenger safety. In *Pattern Recognition. ICPR International Workshops and Challenges*, pages 656–671. Springer International Publishing, 2021.

[16] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-Motion Revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[17] Ewelina Rupnik, Mehdi Daakir, and Marc Pierrot Deseilligny. MicMac – a free, open-source solution for photogrammetry. *Open Geospatial Data, Softw. Stand.*, 2(1):14, dec 2017.

[18] Simon Fuhrmann, Fabian Langguth, and Michael Goesele. MVE - A Multi-View Reconstruction Environment. In Reinhard Klein and Pedro Santos, editors, *Eurographics Workshop on Graphics and Cultural Heritage.* The Eurographics Association, 2014.

[19] OpenDroneMap Authors. Odm - a command line toolkit to generate maps, point clouds, 3d models and dems from drone, balloon or kite images.

[20] Regard3D Authors. Regard3d.

[21] C. Strecha, W. von Hansen, L. Van Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *2008 IEEE Conference on Computer Vision and Pattern Recognition.* IEEE, June 2008.

[22] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vis.*, 60(2):91–110, nov 2004.

[23] Jingming Dong and Stefano Soatto. Domain-size pooling in local descriptors: DSP-SIFT. In *2015 IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 5097–5106. IEEE, jun 2015.

[24] P. F. Alcantarilla, J. Nuevo, and A. Bartoli. Fast explicit diffusion for accelerated features in nonlinear scale spaces. In *British Machine Vision Conf. (BMVC)*, 2013.

[25] Lilian Calvet, Pierre Gurdjos, Carsten Griwodz, and Simone Gasparini. Detection and Accurate Localization of Circular Fiducials under Highly Challenging Conditions. In *2016 IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 562–570. IEEE, jun 2016.

[26] Edwin Olson. AprilTag: A robust and flexible visual fiducial system. In *2011 IEEE Int. Conf. Robot. Autom.*, pages 3400–3407. IEEE, may 2011.

[27] D. Nister and H. Stewenius. Scalable Recognition with a Vocabulary Tree. In *2006 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. - Vol. 2*, volume 2, pages 2161–2168. IEEE.

[28] Jian Cheng, Cong Leng, Jiaxiang Wu, Hainan Cui, and Hanqing Lu. Fast and Accurate Image Matching with Cascade Hashing for 3D Reconstruction. In *2014 IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 1–8. IEEE, jun 2014.

[29] Marius Muja and David G Lowe. Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration. In Alpesh Ranchordas and Helder Araújo, editors, *{VISAPP} 2009 - Proc. Fourth Int. Conf. Comput. Vis. Theory Appl. Lisboa, Port. Febr. 5-8, 2009 - Vol. 1*, pages 331–340. {INSTICC} Press, 2009.

[30] M. Slaney and M. Casey. Locality-Sensitive Hashing for Finding Nearest Neighbors [Lecture Notes]. *IEEE Signal Process. Mag.*, 25(2):128–131, mar 2008.

[31] Martin A Fischler and Robert C Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. Assoc. Comput. Mach.*, 24(6):381–395, jun 1981.

[32] Changchang Wu. Towards Linear-Time Incremental Structure from Motion. In *2013 Int. Conf. 3D Vis.*, pages 127–134. IEEE, jun 2013.

[33] Pierre Moulon, Pascal Monasse, and Renaud Marlet. Global Fusion of Relative Motions for Robust, Accurate and Scalable Structure from Motion. In *2013 IEEE Int. Conf. Comput. Vis.*, pages 3248–3255. IEEE, dec 2013.

[34] Michal Havlena, Akihiko Torii, and Tomáš Pajdla. Efficient Structure from Motion by Graph Optimization. pages 100–113. 2010.

[35] Roberto Toldo, Riccardo Gherardi, Michela Farenzena, and Andrea Fusiello. Hierarchical structure-and-motion recovery from uncalibrated images. *Comput. Vis. Image Underst.*, 140:127–143, nov 2015.

[36] Rajvi Shah, Aditya Deshpande, and P. J. Narayanan. Multistage SFM: Revisiting Incremental Structure from Motion. In *2014 2nd Int. Conf. 3D Vis.*, pages 417–424. IEEE, dec 2014.

[37] H. Hirschmuller. Stereo Processing by Semiglobal Matching and Mutual Information. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(2):328–341, feb 2008.

[38] Xing Mei, Xun Sun, Mingcai Zhou, Shaohui Jiao, Haitao Wang, and Xiaopeng Zhang. On building an accurate stereo matching system on graphics hardware. In *2011 IEEE Int. Conf. Comput. Vis. Work. (ICCV Work.*, pages 467–474. IEEE, nov 2011.

[39] Jose Luis Blanco and Pranjal Kumar Rai. nanoflann: a C++ header-only fork of FLANN, a library for nearest neighbor (NN) with kd-trees. https://github.com/jlblancoc/nanoflann, 2014.

[40] Bruno Lévy and Alain Filbois. Geogram: a library for geometric algorithms. In *Int. Conf. Adapt. Model. Simul.*, Nantes, France, 2015. CNIME.

[41] Michal Jancosek and Tomas Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. In *CVPR 2011*, pages 3121–3128. IEEE, jun 2011.

[42] Michal Jancosek and Tomas Pajdla. Exploiting Visibility Information in Surface Reconstruction to Preserve Weakly Supported Surfaces. *Int. Sch. Res. Not.*, 2014:1–20, aug 2014.

[43] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(9):1124–1137, sep 2004.

[44] Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérome Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph.*, 21(3):362–371, jul 2002.

[45] Peter J. Burt and Edward H. Adelson. A multiresolution spline with application to image mosaics. *ACM Trans. Graph.*, 2(4):217–236, oct 1983.

[46] Adam Baumberg. Blending Images for Texturing 3D Models. *Proc. Br. Mach. Vis. Conf.*, 2003.

[47] Cedric Allene, Jean-Philippe Pons, and Renaud Keriven. Seamless image-based texture atlases using multi-band blending. In *2008 19th Int. Conf. Pattern Recognit.*, pages 1–4. IEEE, dec 2008.

[48] Simone Gasparini, Fabien Castan, and Yann Lanthony. Buddha dataset, January 2017. https://github.com/alicevision/dataset_buddha.

[49] Richard dataset. pi3dscan. https://www.pi3dscan.com/index.php/download/item/example-model-richard.

[50] Paul E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques - SIGGRAPH '97.* ACM Press, 1997.

[51] M.D. Grossberg and S.K. Nayar. Modeling the space of camera response functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10):1272–1282, October 2004.

[52] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "GrabCut". *ACM Transactions on Graphics*, 23(3):309–314, August 2004.

[53] Matthew Brown and David G. Lowe. Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision*, 74(1):59–73, December 2006.