PIXEL VR: Optimizing Photogrammetric Datasets for Standalone VR

Vasili Manfredi[1], Cecilia Maria Bolognesi[2], Stephen Fai [3]

[1,2] Dept. of Architecture, Built environment and Construction engineering, Politecnico di Milano, 20133 Milano, Italy – vasili.manfredi@polimi.it, cecilia.bolognesi@polimi.it
[3] Carleton Immersive Media Studio, Carleton University, 1125 Colonel By Drive Ottawa, Canada – sfai@cims.carleton.ca

Keywords: Photogrammetry, Virtual Reality, 3D Optimization, Cultural Heritage, Open-Source, Automation.

Abstract

Photogrammetric models are increasingly employed for heritage documentation, education, and interactive visualization. However, their complexity and size limit in their applicability on standalone Virtual Reality (VR) devices or low-end machines, which typically operate under significant hardware constraints. This research addresses these limitations through the development of an automated optimization workflow implemented as a Blender Python script. The proposed pipeline integrates a series of processes, remeshing, decimation, UV unwrapping, and texture baking, to significantly reduce polygon count while preserving visual fidelity. Case studies have been retrieved using open-access datasets and original surveys from the Carleton Immersive Media Studio (CIMS) and demonstrate polygon reductions exceeding 99% with minimal visual degradation, enabling real-time visualization on limited hardware. The study emphasizes accessibility and replicability by exclusively utilizing open-source and or free to use software, allowing a scalable, cost-effective solution for immersive cultural applications.

1. Introduction

Photogrammetry has become a widely used technique in the digitization of real-world assets, particularly within the domains of archaeology, architecture, and cultural heritage. It enables the creation of high-fidelity 3D models suitable for various applications such as digital preservation, education, and virtual storytelling. Despite the high level of realism achievable, the widespread use of photogrammetric models in immersive environments is currently limited by the size and complexity of the resulting datasets. These dense models often require substantial computational power, advanced 3D processing knowledge, and time-consuming optimization processes, making them inaccessible to users.
As immersive technologies become more integrated into cultural communication and public engagement strategies, there is an urgent need for streamlined pipelines that transform raw photogrammetric data into light, real-time-ready assets. This research aims to automate and standardize the optimization workflow, leveraging Blender's Python scripting capabilities and integrating external open-source tools such as Instant Meshes. The goal is to create a scalable, hardware-efficient workflow capable of drastically reducing polygon counts without compromising photorealism.
To ensure both reproducibility and versatility, the research employs datasets with diverse characteristics, in terms of scale, acquisition methods, and photographic quality, from small objects to entire buildings. All processes are conducted using accessible, open-source software on mid-range hardware.

2. Dataset and processing

In order to ensure methodological consistency, define best practices, and evaluate different processing scenarios, the research uses open-access datasets and original photogrammetric surveys conducted by the Carleton Immersive Media Studio (CIMS), Carleton University, Canada. The selected case-studies were chosen to embrace different variables, including differences in dataset size, image resolution, acquisition techniques, and photogrammetric hardware configurations.
All datasets were processed using the same hardware setup to eliminate computational differences. A DELL XPS 15 portable workstation was used, equipped with an Intel i7-10750H CPU, an NVIDIA GeForce GTX 1650 Ti GPU, and 16 GB of DDR4 RAM.
The processing software was RealityScan 1.5 (formerly known as Reality Capture, developed by Epic Games, 2025).
Five datasets were analyzed, four retrieved from Epic Games' open-access repository and one provided by the CIMS Laboratory. The first dataset, “Cottage Model” consists of 197 images captured with a SONY ILCE-7M3 camera at 6000×4000 px, using a 24 mm focal length. It was selected for its medium image count and balanced acquisition scale. The “High Relief” dataset includes only 14 images, taken with a compact Canon DIGITAL IXUS 85 IS (2816×2112 px, 68.3 mm focal length), and was chosen to assess reconstruction outcomes from minimal, low-cost image input.
The “Hammer” dataset comprises 190 images captured with a Canon EOS 6D (5472×3648 px, 49.3 mm focal length), providing a comparative mid-sized dataset. The “Waitangi Canoe” dataset, includes 715 high-resolution images (7360×4912 px); although both the camera model and focal length are unspecified, it was selected for its dense coverage and image quality. The Bytown Museum dataset, which was provided by the CIMS Laboratory, features 1178 images captured with a SONY ILCE-7RM3 (7968×5320 px, unknown focal length) and aerial drone. The subject is a heritage building located near the Rideau Canal in Ottawa, it was selected for the high quality of the photographic material.
All datasets were processed using identical computational resources, allowing a clear evaluation of the photogrammetric outcomes.

Figure 1. “Hammer” dataset mesh creation into Reality Scan.

DATASET NAME	IMAGES	ALIGNMENT RESULT	IMAGE RESOLUTION	HARDWARE
COTTAGE MODEL	197	197/197	6000×4000px	SONY ILCE-7M3
HIGH RELIEF	14	14/14	2816×2112px	Canon DIGITAL IXUS 85 IS
HAMMER	190	190/190	5472×3648px	Canon EOS 6D
WAITANGI CANOE	715	677/715	7360×4912px	Unkwown
BYTOWN MUSEUM	1178	1162/1178	7968×5320	SONY ILCE-7RM3

Table 1. Concise table of the Dataset used.

3. Proposed Solution

The proposed solution aim to expand current techniques of photogrammetric dataset for Immersive Technologies asset creation. The main focus was on automatization of mesh optimization processes, that could allow time reduction processes and easier asset creation.

3.1 Image Processing and mesh creation

Prior to importing the image datasets into the photogrammetric processing software, a pre-processing phase was carried out to enhance the input quality and to ensure consistency of the case studies.
For each dataset, the image-sets have been batch edited, with adjustments made to image histogram parameters such as highlights and shadows. This step aimed to improve the contrast and reveal additional surface detail, particularly in outdoor scenes where lighting conditions may introduce reflective noise or exposure imbalances. The process contributed to a more stable and faster mesh reconstruction and reduced light-related artifacts during geometry generation.
Following the pre-processing phase, the edited images were imported into Reality Scan 1.5 (Epic Games, 2025) for 3D reconstruction.
In the latest software version, an AI-based masking tool is available to automatically isolate the survey subject from the background. While, it was not utilized in the initial processing phase, this feature was later tested; however, no significant improvements were observed in the tested datasets.
Further analysis is required to assess its effectiveness in other scenarios.
For each dataset, the photogrammetric workflow began with the alignment process, which computes a sparse point cloud used to evaluate the coverage and quality of the photographic acquisition. Once alignment was successfully achieved, the mesh calculation phase was initiated.
Reality Scan offers multiple preset options for mesh generation: Draft, Medium Detail, and High Detail. In order to balance computational performance with geometric accuracy, each dataset was first processed using the Medium Detail preset, followed by a High Detail iteration for comparison.
The Medium Detail preset typically achieve high-quality reconstructions with a relatively reduced polygon count and faster processing time, though in some cases it results in smoother surfaces and a loss of fine geometric features. The High Detail setting produces significantly denser meshes, ranging from 4 to 10 times more polygons, thereby preserving surface complexity and reducing noises across the model at a higher computational cost.
Once mesh generation was completed, each model was UV-unwrapped using the integrated Unwrap tool.
This automatic function allows the user to generate UV layouts optimized for the software’s texturing process. Based on the UV layout and surface definition, high-resolution 8K textures were subsequently applied directly within Reality Scan, resulting in photorealistic outputs that preserved both colorimetric fidelity and spatial coherence.

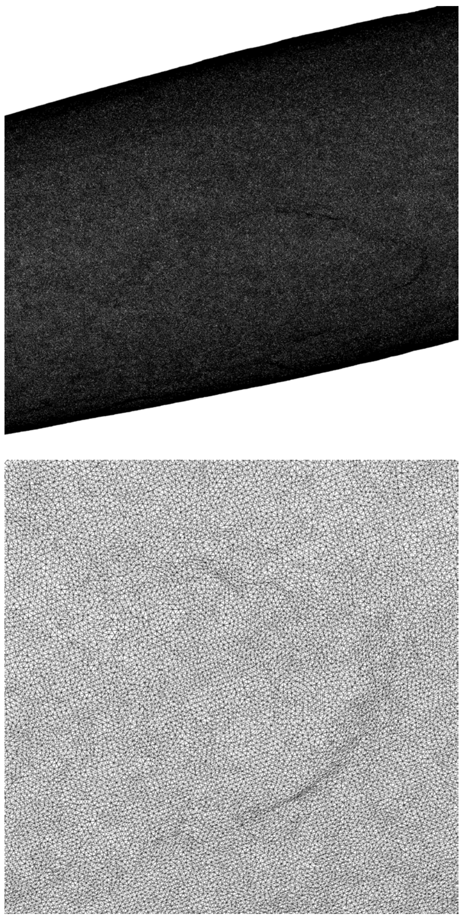


Figure 2. Geometrical density of the High Detail preset reconstruction into Reality Scan.

To ensure reproducibility and facilitate comparative analysis, the same processing parameters were systematically applied to all datasets.

Folder structures and naming conventions were also standardized in preparation for the subsequent organization of textures and mesh files in later stages of the workflow.

It was decided to maintain the meshes output as clean as possible, without polygon reduction inside Reality Scan to compare the manual optimization process and the automated one.

DATASET NAME	POLY COUNT MEDIUM PRESET	POLY COUNT HIGH PRESET	TEXTURES SIZE	.FBX SIZE
COTTAGE MODEL	5,508,998	44,071,984	62,354 KB	126,727 KB
HIGH RELIEF	1,381,244	5,753,745	58,612 KB	203,182 KB
HAMMER	2, 241,584	9,238,698	42,088 KB	207,098 KB
WAITANGI CANOE	24,050,972	192,407,782	84,847 KB	1,021,474 KB
BYTOWN MUSEUM	38,125,000	305,000,000	320,936 KB	364,091 KB

Table 2. Dataset processing output data.

3.2 Manual mesh processing

For the post-processing and optimization of the high-resolution photogrammetric meshes, Blender 4.3 has been used due to its extensive suite of tools for three-dimensional data manipulation, ranging from basic geometric operations to advanced scripting functionalities. Its embedded support for Python scripting provides an interface for the automation of mesh editing and texturing processes, which are conventionally executed manually by expert users.

Traditionally, the optimization of complex photogrammetric meshes involves a highly customized, operator-dependent workflow, requiring substantial computational resources and advanced 3D modelling expertise. This manual pipeline typically includes cleaning operations, remeshing, topology enhancement, UV unwrapping, and texture baking. As a result, the process is a time-consuming and often unsustainable when dealing with multiple datasets or when targeting extremely light outputs for fast and low-cost immersive visualization.

In this research is proposed a the development of a semi-automated pipeline that mimics the manual optimization logic through a Python-based algorithm executed within Blender’s scripting environment. The algorithm replicates the conventional optimization stages, such as geometry refinement, non-manifold edge removal, decimation of redundant details, and mesh preparation for texture transfer.

For comparative purposes, both workflows, the manual (“tailor-made”) and the scripted approach, are discussed. The manual pipeline begins with the import of high-poly meshes, generated in Reality Scan, into Blender using the .FBX format.

This format ensures the preservation of essential metadata, such as spatial scale, potential georeferencing, and embedded textures. The initial phase involves cleaning the raw mesh, which may include removing loose geometry, non-relevant context, isolated vertices, and non-manifold elements. To ensure a non-destructive approach, the original model is duplicated before any editing. Subsequently, a base topology is generated using Instant Meshes, an open-source software capable of producing quad-dominant remeshing from triangulated input. The resulting mesh, exported in .obj format, is re-imported into Blender for further refinement. The re-imported mesh undergoes manual topological correction to delete artifacts, irregular face distributions, and surface discontinuities.

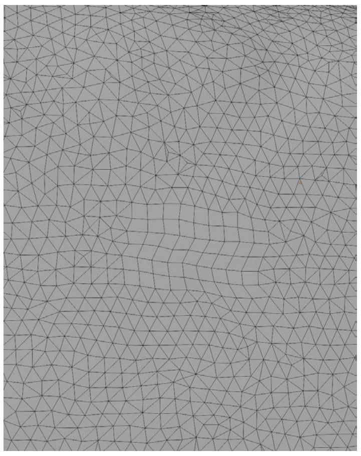


Figure 3. Example of manual retopology

At this stage, the mesh can be remeshed or decimated using Blender’s embedded tools. To ensure effective optimization, the mesh is segmented into topologically meaningful clusters. Each cluster is then individually processed, either through global or local decimation techniques, based on its geometric complexity and relevance to the final visualization. This process allows for controlled polygon reduction while preserving critical details.

Once the topological optimization is finalized, the mesh is UV unwrapped using a UDIM-based UV layout. The UDIM system enables the distribution of UV islands across multiple tiles (1001, 1002, etc.), enhancing texture resolution without compromising editing flexibility. Following UV unwrapping, the baking process is executed. This operation transfers high-frequency visual information, such as color (diffuse) and normal, from the original high-poly mesh to the optimized low-poly version. The baking process ensures that despite the reduction in polygon count, the visual quality is retained

through texture maps, which simulate the surface complexity of the original geometry.
The outcome is a highly optimized 3D model with reduced computational demands, suitable for real-time immersive environments.

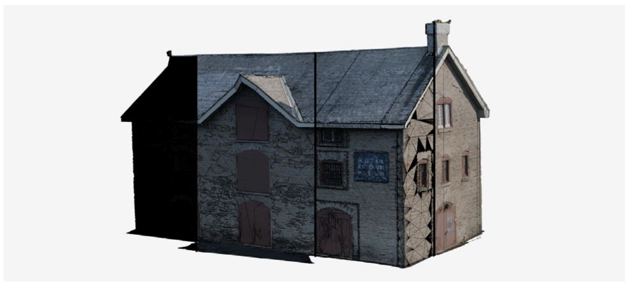


Figure 4. “Bytown Museum” dataset manually processed by using different techniques

3.3 Processing Algorithm

To enhance the most complex and time-consuming processes the proposed workflow integrates a Python-based semi-automated algorithm developed in Blender 4.3, combining remeshing and decimation functionalities to restructure and simplify raw 3D models. The remeshing operation is conceptualized as a preprocessing phase, targeting meshes imported from photogrammetric tools in highly triangulated formats (.FBX or .OBJ). The algorithm triggers a custom export function that extracts the duplicated high-poly mesh and saves it as a standalone .obj file, explicitly prepared for remeshing via an external software, Instant Meshes. The operator launching Instant Meshes is embedded into the Blender interface, providing a seamless bridge between the environments. The automated optimization is conceptualized in between of the mesh creation phase and the final output in the visualization engines. The tool work on highly triangulated photogrammetric output, reading the main 3D formats as .FBX, .OBJ and glTF. Once the Python script has been run inside Blender it gives to the user multiple tools for automated optimization. The optimization tools are structured to work consequentially until the final optimized model. The first tool optimize the output from the photogrammetric process, as explained in the manual optimization, it detect and clean any extra or non manifold geometry and prepares the mesh to be remeshed. For the remeshing stage has been decided to rely on an external software: Instant Meshes, which remeshing algorithm is very suitable for this kind of optimization. The operator is embedded into Blender’s UI interface to provide a seamless bridge between the two environments. Instant Meshes utilizes curvature-aware quadrangulation algorithms to convert triangulated data into structured quad-dominant meshes, that are very suitable for the next steps to be recursively decimated.

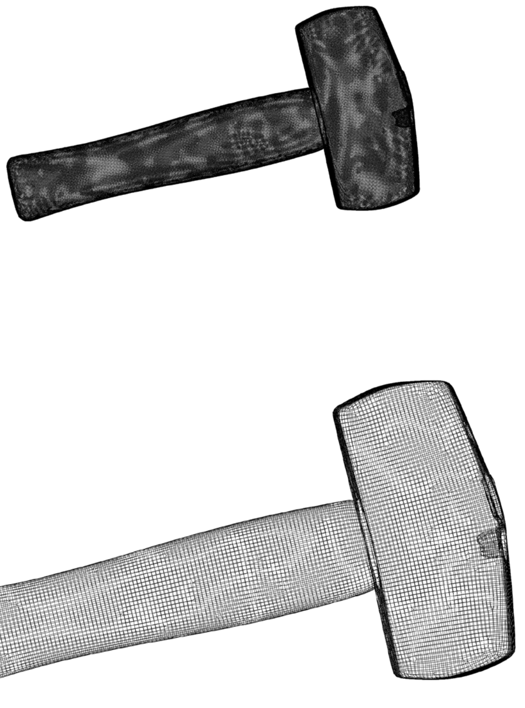


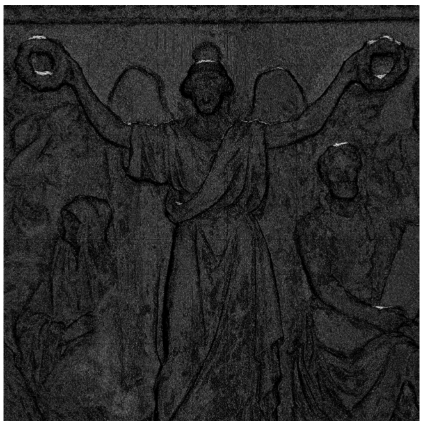
Figure 5. Sequence of retopology of a mesh using the Python script.

After processing, the mesh is re-imported by using the import operator embedded into Blender’s interface and automatically assigned to a dedicated "Optimized" collection within Blender, ensuring a clear separation from the original high-density data. Following remeshing, the algorithm proceeds with a multi-layered decimation phase, designed to reduce polygon count while preserving visual integrity by recursively decimation. Implemented through Blender’s native Decimate modifier, the decimation logic follows both manual and adaptive paths:

• Manual Ratios: The interface provides direct access to defined decimation presets tailored for immersive applications: Cinematic (0.5), Game (0.1), and Data (0.01) which are applied iteratively on mesh duplicates.

These variants are automatically sorted into labeled sub-collections for further evaluation and for keeping possibilities to went back.

• Curvature-Based Vertex Group Exclusion: To preserve morphological fidelity in complex or rounded regions, the algorithm has a vertex selection that analyzes local edge density. Vertices linked to highly detailed topological regions are assigned to a protected group named low_decimation_detail.



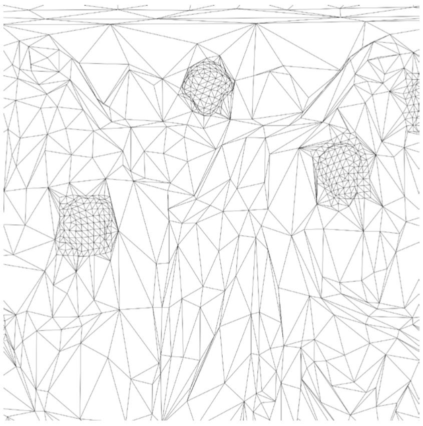


Figure 6. Curvature-Based algorithm behaviour on the "High relief" dataset.

When enabled, the decimation process excludes this vertex group from simplification, preserving details where needed most such as carved surfaces, ornamentation, or anatomical details. This segmentation based approach supports a layered simplification model, where redundant planar faces are collapsed while high-detail segments remain intact. The result achieved really high compression ratios with minimal perceptual degradation. Prior to and after decimation, the system performs a mesh consistency check to eliminate non-manifold edges, apply scale transformations, and ensure uniform normal orientation. These operations are executed via embedded _bmesh routines that scan for topological anomalies and clean the dataset accordingly. Following the remeshing and decimation stages, the optimized meshes are prepared for texture transfer by a UV unwrapping process. This step is critical to enable the projection of high-resolution visual data (diffuse and normal maps) from the photogrammetric source onto the new optimized geometry.

The method adopted in this research leverages both traditional UV unwrapping techniques and an advanced UDIM tiling approach, depending on the complexity and scale of the subject. For standard unwrapping, the algorithm uses Blender's Smart UV Project operator to flatten the mesh into two-dimensional UV coordinates. This method is particularly effective for irregular or organically shaped models, offering a balance between minimization of distortion and automation. The operator parameters are preset to a conservative angle limit of 66°, with an island margin of 0.03 units, ensuring minimal overlap between UV islands. Subsequently, the UV islands are automatically packed using Blender's _pack_islands operation, which fill the available UV space as efficiently as possible. This standard layout is typically applied when a single 2K or 4K texture suffices for visual fidelity.

For complex models or heritage datasets where visual detail must be preserved at high fidelity, the algorithm enables UDIM-based unwrapping. In this approach, the UV layout is distributed across multiple tiles (1001 to 1006), each representing an independent 2D texture canvas. This allows for substantially higher effective resolution without exceeding the maximum dimensions of a single texture map.

The UDIM workflow is implemented by dividing the UV space into layout and unwrapped different mesh clusters to each tile. The algorithm provides a visual interface for selecting resolution presets (1K, 2K, 4K, 8K) and ensures that each UDIM tile is filled with proportionally scaled UV islands to optimize texel density. This system enables multi-tile baking while maintaining the resolution budget per tile, which is essential when targeting high-fidelity rendering in immersive environments taking account of texture memory limits.

With the UV layout finalized, the system prepares the mesh for baking:

• Create a new material set with a standard Principled BSDF shader.

• Generate two blank image textures: one for the diffuse map and one for the normal map. The user can set the resolution and enable 32-bit float precision for the normal map to ensure smooth shading and compatibility with downstream rendering engines.

• Connect the textures to the shader for preview purposes.

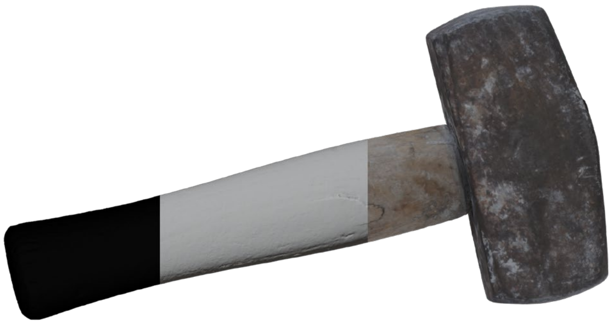


Figure 7. Composition of the "Hammer" dataset.

The baking procedure is executed using Blender's Cycles rendering engine, simulating the high-poly source model's surface detail onto the UV layout of the low-poly optimized mesh. The script first bakes the diffuse component, followed by the normal map. The baked textures are then saved in a writable path using a systematic naming convention, ensuring compatibility with external rendering engines or game engines. A critical implementation detail in the baking process relates to the normal map axis orientation. Blender's internal normal baking system uses a +Y tangent space convention by default, whereas Unreal Engine requires normal maps in -Y (OpenGL-style) format for correct shading behavior. To address this, the baking algorithm explicitly sets the normal map orientation to -Y prior to the bake operation, ensuring visual compatibility and preventing shading artifacts. The result is a set of low-poly meshes augmented with high-resolution textures that replicate the visual characteristics of the original survey that using limited computational resources. Once the geometry and texturing stages have been completed, the final step in the pipeline involves preparing the optimized mesh and associated textures for integration into real-time visualization platforms, such as Unreal Engine. The algorithm includes an automated export operator which allows the user to export the optimized mesh as an .obj, .fbx or glTF file using a predefined naming convention (e.g., ModelName_Optimized.obj). This export also ensures that the model is stored in a clean, engine-ready state: transformations are applied, normals are recalculated, and detached geometry is removed. The textures are baked and saved in image formats (typically .png), stored alongside the geometry file to preserve the texture-mesh linkage. To maintain alignment between texture coordinates and shading information, the UV layout generated during the unwrapping stage is embedded in the exported file. For UDIM-based workflows, the exported mesh retains tile indexing to allow correct texture assignment in engines that support the UDIM standard (eUnreal Engine). To ensure consistency and scalability, the final assets are structured into organized directories. Each optimized mesh is placed in a folder along with its corresponding diffuse and normal maps. When using UDIM tiles, each tile image is named with its respective index (Texture_1001.png, Texture_1002.png, etc.). These assets are then ready for import into real-time engines such as Unreal Engine 5, where additional features like Virtual Textures can further enhance rendering performance and visual quality. Because the exported assets maintain consistent naming, scale, and coordinate alignment, the transfer process is streamlined and does not require manual adjustments.

3.4 Results

The optimization process was tested on five datasets of varying complexity and size. The evaluation focuses on polygon count reduction, texture size compression, .FBX file size, and rendering time improvements. All results were obtained on a Dell XPS 15 9500 with 16 GB RAM, Intel i7-10750H, and GTX 1650 Ti GPU.
High Relief: Reduced from 5,753,745 to 4,036 triangles (99.93% reduction). Texture size reduced from 203,182 KB to 5 KB (99.99%). .FBX file size dropped from 58,612 KB to 37,702 KB (35.67%). Rendering time halved from 7.13 to 3.22 minutes (54.83% faster).
Cottage Model: Optimized from 44,071,984 triangles to 3,349, achieving a 99.94% reduction. Texture size dropped from 126,727 KB to 254 KB (99.79% reduction). .FBX size decreased by 52.71% (from 62,354 KB to 29,482 KB). Rendering time dropped from 5.37 to 1.50 minutes (72.06% faster).
Hammer: Reduced from 9,238,698 to 2,894 triangles (99.97%). Texture file size decreased from 207,098 KB to 137 KB. While .FBX size slightly increased from 42,088 KB to 48,000 KB (due to embedded maps), rendering time dropped from 1.25 to 0.39 minutes.
Waitangi Canoe: Original face count of 305,000,000 was reduced to 5,508,998 (98.19% reduction). Texture size was compressed from 819,200 KB to 92,160 KB (88.75% reduction). Rendering time was reduced from 12.45 minutes to 4.11 minutes (66.98%).
Bytown Museum: Original dataset with 1,178 images. Face count went from 98,157,231 to 14,231 (99.99% reduction). Texture memory dropped from 354,812 KB to 1,124 KB. .FBX file shrunk from 187,350 KB to 12,430 KB (93.36% reduction). Rendering time improved from 10.85 to 2.76 minutes (74.56%).

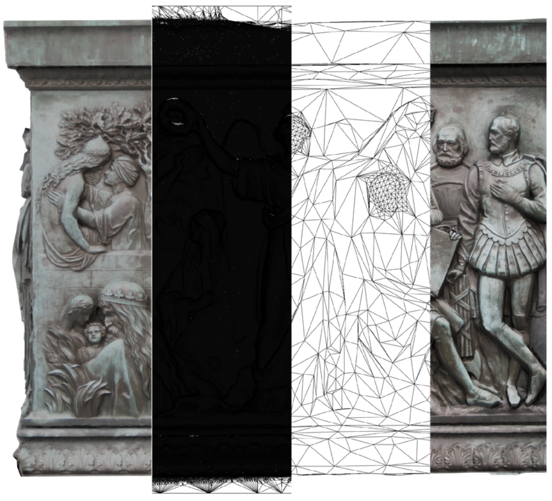


Figure 8. Optimization process of the "High Relief" dataset.

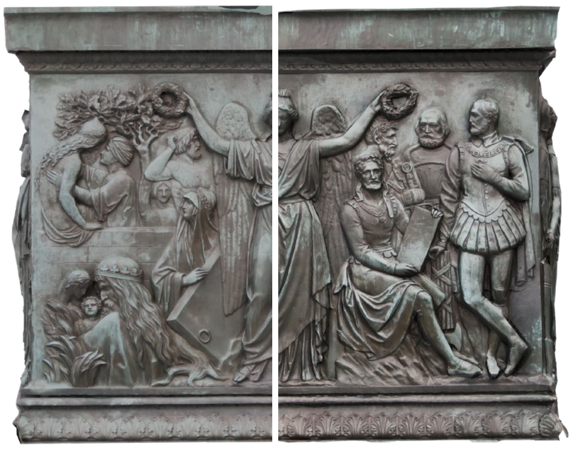


Figure 9. On the left side the original output of "High relief" dataset, on the right side the optimized one.

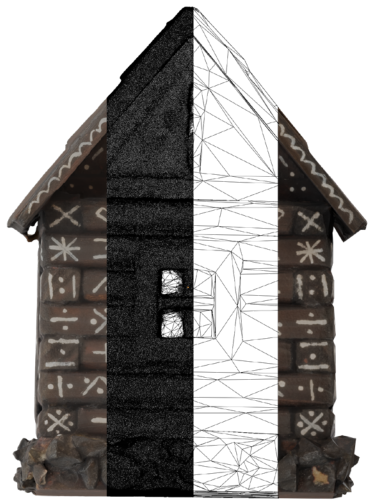


Figure 10. Optimization process of the "Cottage" dataset.

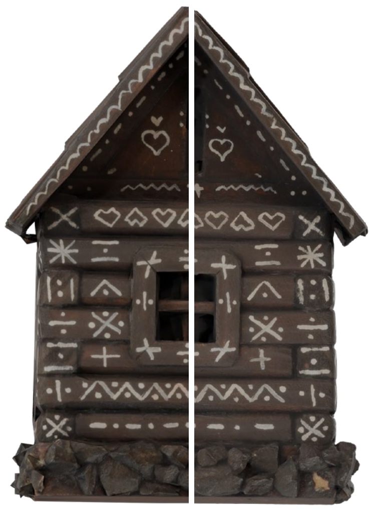


Figure 11. On the left side the original output of "Cottage" dataset, on the right side the optimized one.

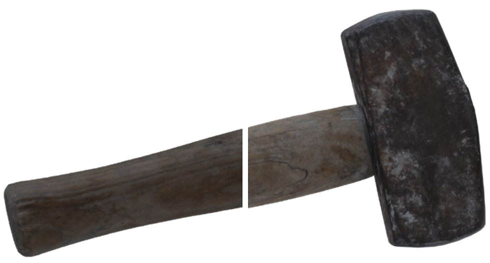


Figure 12. On the left side the original output of "Hammer" dataset, on the right side the optimized one.

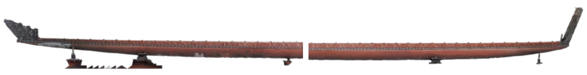


Figure 12. On the left side the original output of "Waitangi Canoe" dataset, on the right side the optimized one.

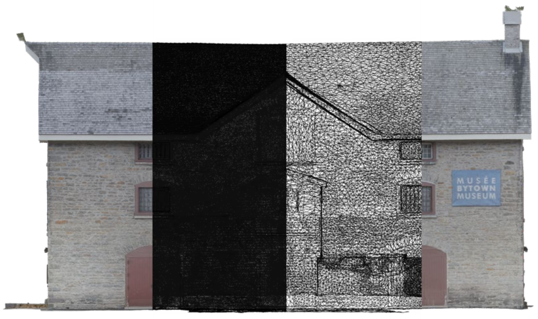


Figure 13. Optimization process of the "Bytown Museum" dataset.

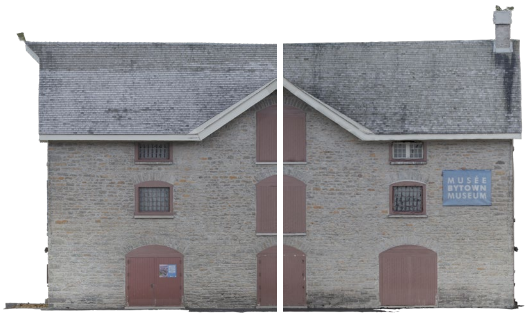


Figure 14. On the left side the original output of "Bytown Museum" dataset, on the right side the optimized one.

PROCESSING HARDWARE: DELL XPS 15 9500	FACE COUNT	TEXTURE SIZE (D+N)	.FBX SIZE	RENDERING TIME PATH TRACING 2K_DENOISED
COTTAGE	44,071,984	62,354 KB	126,727 KB	5.37 Minutes
COTTAGE_OPTIMIZED	3,349	29,482 KB	254 KB	1.50 Minutes
REDUCTION PERCENTAGE	99,99%	52.72%	99.80%	72.06%
HIGH RELIEF	5,753,745	58,612 KB	203,182 KB	7.13 Minutes
HIGH RELIEF_OPTIMIZED	4,036	37,702 KB	5 KB	3.22 Minutes
REDUCTION PERCENTAGE	99,93%	35.67%	99.99%	54.83%
HAMMER	9,238,698	42,088 KB	207,098 KB	1.25 Minutes
HAMMER_OPTIMIZED	2,894	48,000 KB	137 KB	0.39 Minutes
REDUCTION PERCENTAGE	99,97%	-14.04%	99.93%	68.8%
CANOE	192,407,782	84,847 KB	1,021,474 KB	57.57 Minutes
CANOE_OPTIMIZED	27,538	52,800 KB	1,452 KB	1.13 Minutes
REDUCTION PERCENTAGE	99,99%	37.77%	99.85	98.03%
BYTOWN MUSEUM	305,000,000	320,936 KB	364,091 KB	7,21 Minutes
BYTOWN MUSEUM_OPTIMIZED	1,400	24,000 KB	466 KB	1.39 Minutes
REDUCTION PERCENTAGE	99,99%	92.52%	99.87	80.72%

Table 3. Synthesis of the Dataset optimization processes.

4. Conclusion and future research

This study demonstrates the potential of a semi-automated pipeline to optimize photogrammetric datasets for immersive applications.
Through a combination of geometry simplification, curvature-aware segmentation, and texture baking, the workflow achieved significant polygon and file size reductions while maintaining a high level of photorealism. The assets were tested successfully in real-time on standalone VR headsets, demonstrating hardware efficiency and visual fidelity.
The integration of open-source tools ensures that the methodology remains adaptable and replicable by non-experts and institutions with limited budgets. The use of UDIM-based unwrapping, curvature-aware decimation control, and -Y normal orientation increases compatibility with real-time engines such as Unreal Engine.
Future developments will focus on embedding the remeshing operation directly into Blender, replacing dependencies on external tools. Additional researches will be focused on the implementation of more functions to support unexperienced users.

Acknowledgements

This work has been supported by NGI Enrichers , EU-funded project N° 101070125 - Horizon Europe funding.

References

Bekele, M. K., Pierdicca, R., Frontoni, E., Malinverni, E. S., & Gain, J. (2018a). A survey of augmented, virtual, and mixed reality for cultural heritage. Journal on Computing and Cultural Heritage, 11(2), 7. https://doi.org/10.1145/3145534

Blender Development Team, 2023. Version 4.3. (November 19, 2024).

Caetano, A., & Sra, M. (2022). ARfy: A Pipeline for Adapting 3D Scenes to Augmented Reality. UIST 2022 Adjunct - Adjunct Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology. https://doi.org/10.1145/3526114.3558697

Chiabrando, F., Donato, V., Lo Turco, M., & Santagati, C. (2018). Cultural heritage documentation, analysis and management using building information modelling: State of the art and perspectives. Intelligent Systems, Control and Automation: Science and Engineering, 92, 181–202. https://doi.org/10.1007/978-3-319-68646-2_8

Cignoni, P., Montani, C., & Scopigno, R. (1998). A comparison of mesh simplification algorithms. Computers & Graphics, 22(1), 37–54. https://doi.org/10.1016/S0097-8493(97)00082-4
Gledhill, D., & Novak, M. (2023). A Novel Methodology for the Optimization of Photogrammetry Data of Physical Objects for Use in Metaverse Virtual Environments. 2023 IEEE International Conference on Metrology for EXtended Reality, Artificial Intelligence and Neural Engineering, MetroXRAINE 2023 - Proceedings, 40–45. https://doi.org/10.1109/METROXRAINE58569.2023.10405684

Kersten, T. P., Tschirschwitz, F., Deggim, S., & Lindstaedt, M. (2018). Virtual Reality for Cultural Heritage Monuments – from 3D Data Recording to Immersive Visualisation. https://doi.org/10.1007/978-3-030-01765-1_9

Murphy, C., Carew, P. J., & Stapleton, L. (2022). Towards a Human-Centred Framework for Smart Digital Immersion and Control for Cultural Heritage Applications. IFAC-PapersOnLine, 55(39), 30–35. https://doi.org/10.1016/j.ifacol.2022.12.006

Perry Hobson, J. S., & Williams, A. P. (1995). Virtual reality: A new horizon for the tourism industry. Journal of Vacation Marketing, 1(2), 124–135. https://doi.org/10.1177/135676679500100202

Poux, F., Valembois, Q., Mattes, C., Kobbelt, L., & Billen, R. (2020). Initial user-centered design of a virtual reality heritage system: Applications for digital tourism. Remote Sensing, 12(16), 2583. https://doi.org/10.3390/RS12162583

RealityCapture - 3D Models from Photos and/or Laser Scans. (n.d.). Retrieved March 16, 2025, from https://www.capturingreality.com/

Sample datasets - Capturing Reality. (n.d.). Retrieved March 16, 2025,
,