4. 3D Scans evolution

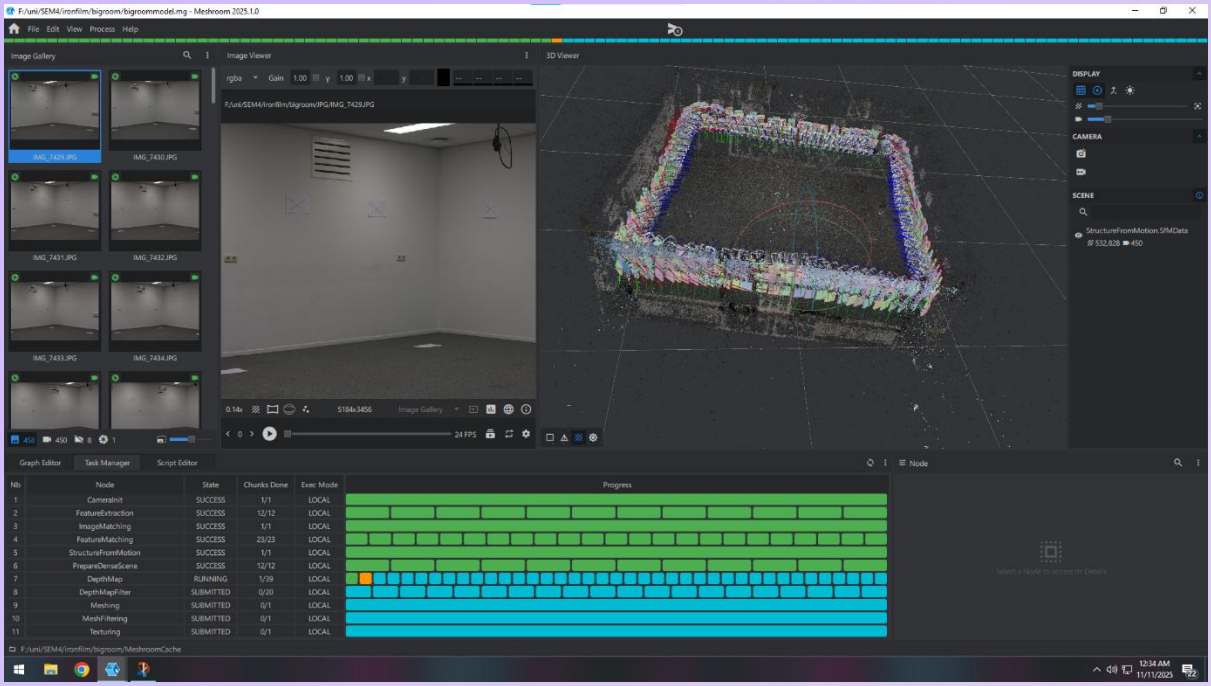
Intro

Started working for this project, my first impression and my first tries were photographing rooms as empty as they could be in the R3-4 building on our campus, since our task was scanning indoor spaces. After doing some research on how to properly photograph, I found a method called orbital scanning, where you set an imaginary point in the room, usually in the middle, and orbit along the walls while always pointing the camera at that center point.

Why orbital method - document

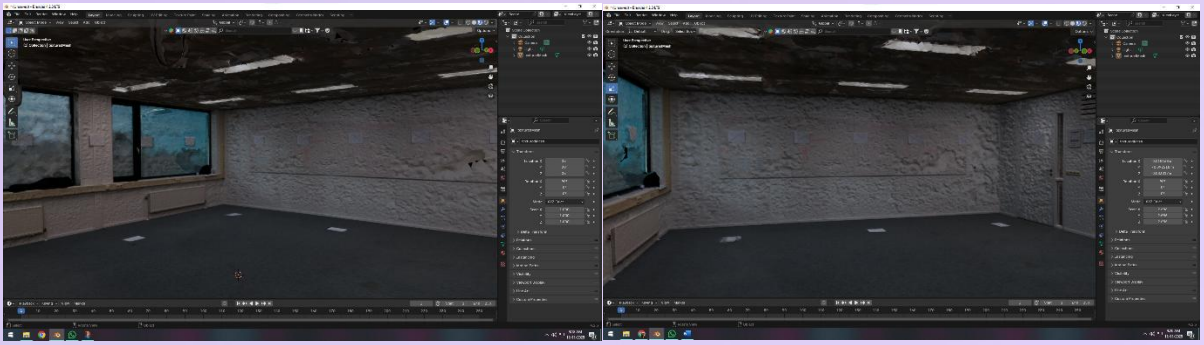
Best practice includes taking the pictures at least 2 different levels (for example 130 cm height and then another at 170 cm), with an overlap in pictures around 70-80%, since this is photogrammetry and the measurements need to be as accurate as possible. That's why we also used hand-drawn markers on paper (X's or circles with an X in them) to help the software map the camera positions easier and measure distances between markers and walls.
For the camera, I used a DSLR Canon 650D, trying to go for an ISO as low as possible, aperture f/8-f/11, shutter 1/60s+, white balance fixed, camera on manual.

Defining the workflow - document



Process & Feedback

The results at first seemed pretty promising. Even with a lot of noise on the walls, we managed to capture the room very well, and small details on the board in the room or a sign next to the door came out almost perfect. So, I kept on trying to improve the method and test daylight vs LED light, switching camera settings and testing different settings in Meshroom, the software I used for renders.

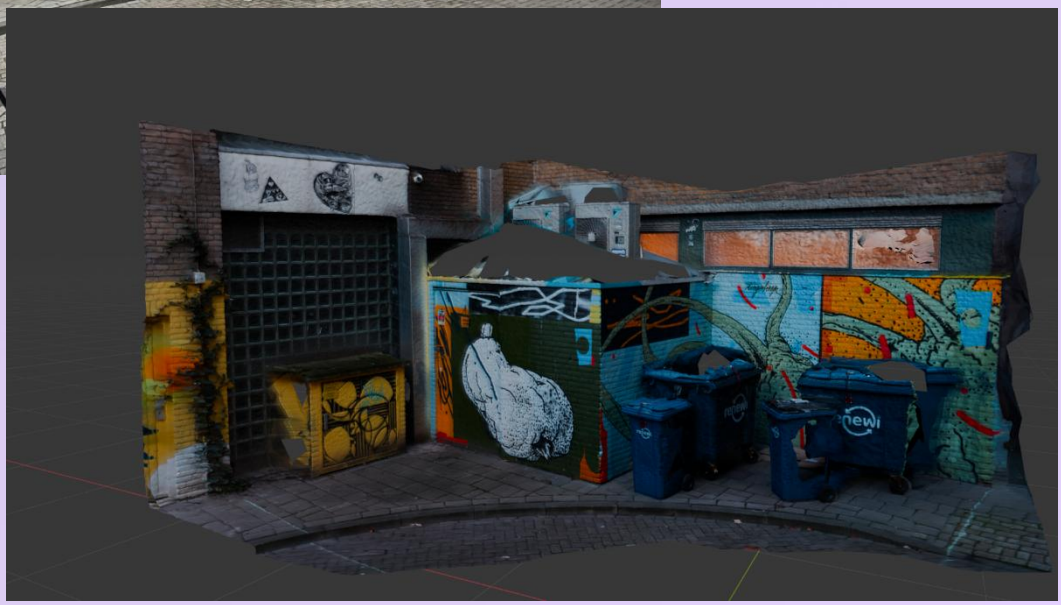


The biggest shock came when me and Razvan, my teammate, went to Pixel Playground to use the powerful PC there to try to render a scan on ultra settings. I was under the impression that the higher the settings, the better the result would be. But seeing the result, which was horrific, I realised that with a bad data set (pictures), upping the settings would just show in even more detail all the flaws.

We wanted to show all these findings and results to the client. We had a meeting and the feedback came:

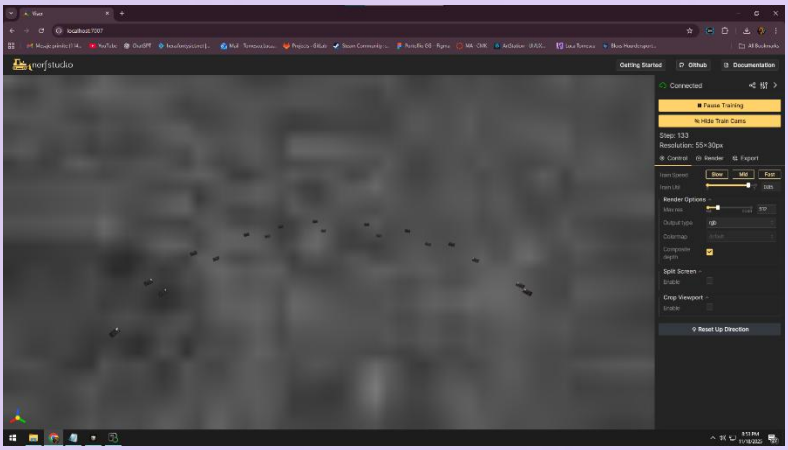
- flat and matte surfaces are very hard to pinpoint in space
- they tend to look like each other, and having fully white walls results in a lot of noise
- we tried using markers to solve that, but the white walls were still a huge problem
- second part of the feedback: look for better rooms/locations with more textured objects/walls (like a church, a library, etc.)

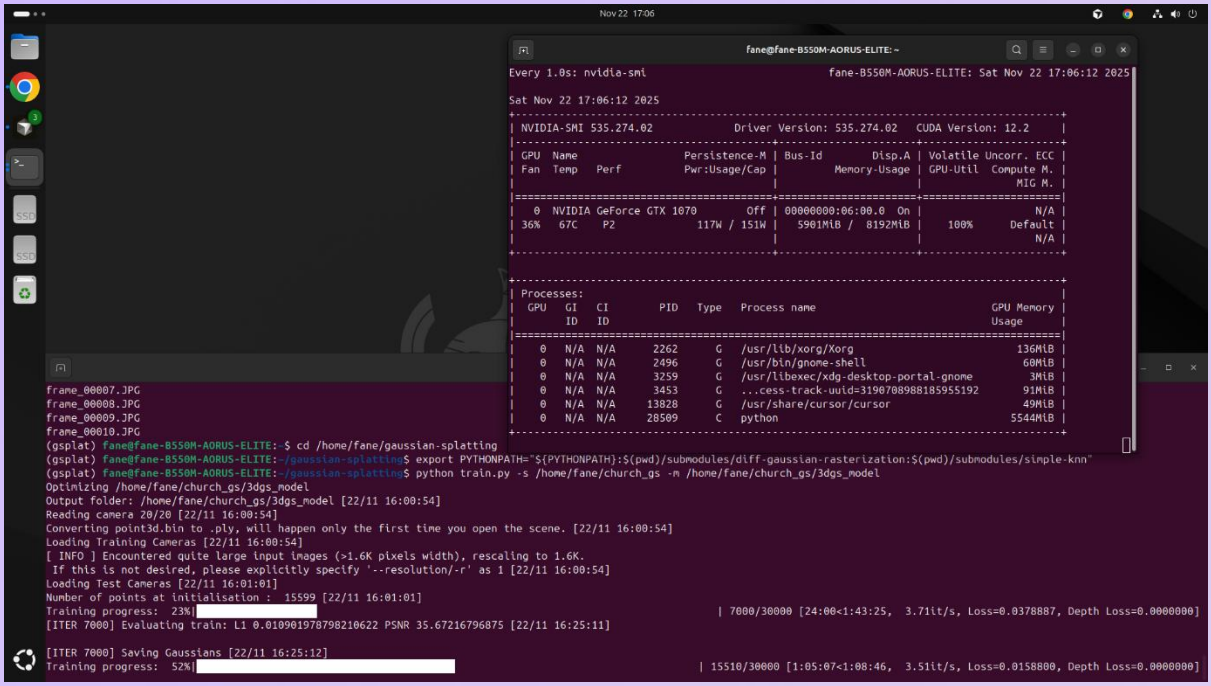
Applying the newly received feedback, we started searching for new locations. We found Achterom street and the church in the city center. We repeated the scan process, and it was time for the render. The results were incredible, nothing like before. They were so clear: textured walls in Achterom, and the church having so many textures on doors, statues and so on. But there was still a problem with matte-color surfaces like the trash bins in Achterom, so it was time to pivot to other technologies.



Going back to the IronFilms presentation, I found NeRF and Gaussian Splatting. Even though in the beginning and in the first meeting with our client we showcased a MoSCoW plan and included photogrammetry as the only MUST, our mission was to get a photorealistic, virtual-production-ready set. And what we had so far was not there yet.

So, I started trying to work with NerfStudio, which ended up failing because I didn't have a powerful enough rig to run a normal Windows Gaussian Splat. So, I had to find a way to make this work. I researched and found a loophole through Linux + NVIDIA drivers that allowed me to run a Gaussian Splat render. After a lot of tries and a lot of work getting everything to run on Linux (and using Linux for the first time in my life), I was finally able to produce our first Gaussian Splat scans.





This is not a picture, this is a gaussian splat render:

Reflection

This whole experience taught me a lot, both technically and in terms of workflow. I learned how important the actual photo-taking process is for photogrammetry and how much overlap, texture, and lighting influence the result. Testing different settings and comparing noisy indoor scans with much cleaner outdoor or textured environments made me understand why choosing the right location matters as much as the software.

The process also showed me how essential client feedback is. Changing locations based on their input directly improved the quality of the scans, and it helped me see the value of adapting quickly instead of sticking to the first plan.

Moving from photogrammetry to NeRF and Gaussian Splatting pushed me to figure out new tools fast, troubleshoot problems, and even set up Linux for the first time. Managing to get a Gaussian Splat to work after all the failed attempts gave me confidence that I can handle technical challenges, even when the hardware or software is limiting.

Overall, this project helped me get better at experimenting, improving methods, and finding solutions when something doesn’t work the first time.