

Protocol – Noam Hartmann

Using MoGe with Cuda version 12.8

Python 3.13.3

Disclaimer!

MoGe uses at large Cuda compatible Pytorch, the problem is Pytorch has only compatibility to 11.8, this makes it very difficult to get it to run as the pre-release version of Pytorch for Cuda 12.8 is not compatible with MoGe.

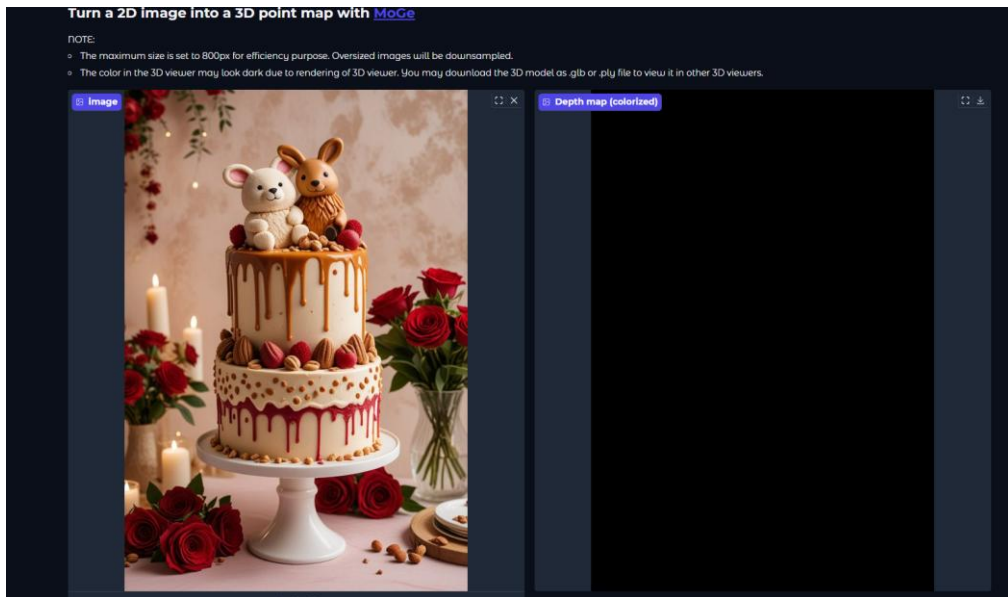
This can be avoided by either trying to downgrade the Cuda which did not work in my case due to software issues or use the pytorch version that only uses the CPU instead of Cuda, which comes with the big downside of being very slow as proven by the simple fact that as the time of writing the code currently runs for 20 minutes and is still ongoing.

This of course makes it very difficult to actually do any testing regarding to the workshop, as well as not being able to work with the in the workshop provided 3D cameras, so this document is going to be very theoretical!

Testing with/against the Azure Kinect

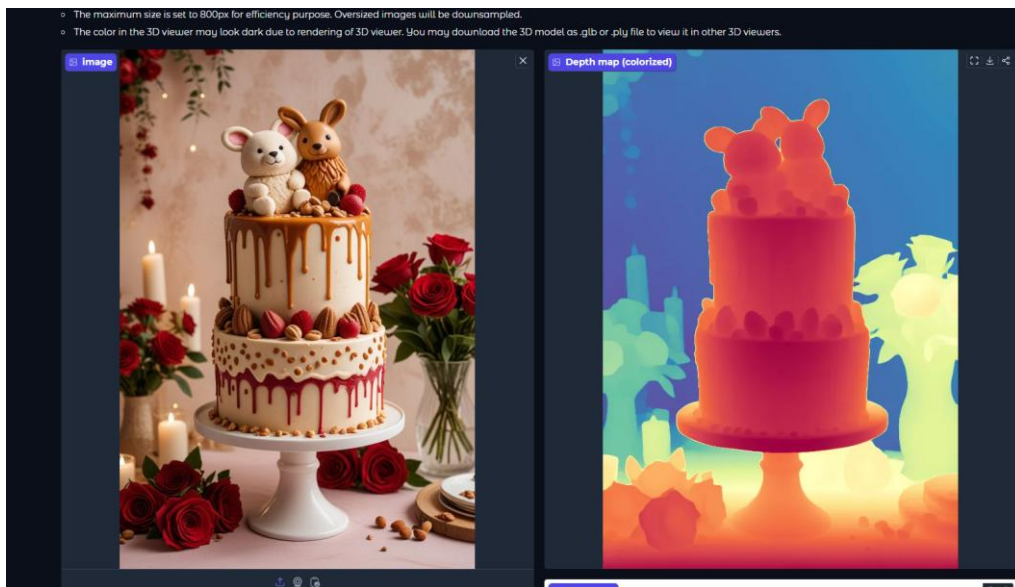
- Different conditions:
 - Trough glass
 - Images further away or close by
 - Etc.
- Testing MoGe's processing speed
- Compare MoGe against other AI models

MoGe within its package has a built-in app that can be used to process and download images this can be modified and run but does unfortunately not work with Cuda 12.8.



Picture run through Built-In app.py in package

However MoGe offers a website in which one might test images the same way it would through the local application, this being said I was able to make some rudimentary test with other images.



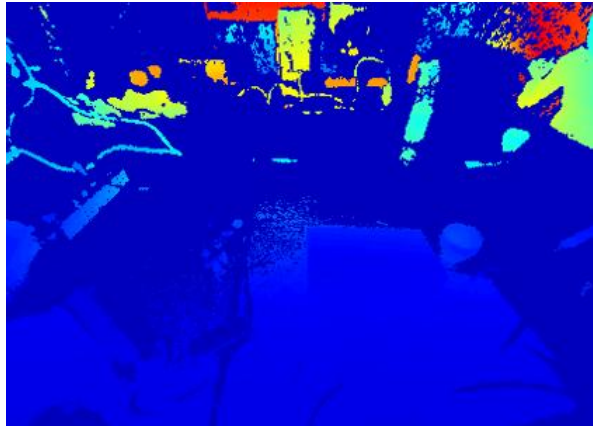
How it should look like – Source: <https://huggingface.co/spaces/Ruicheng/MoGe>



P1: Picture taken through Azure Kinect (by Marcel Ritter)



P2: Screenshot taken from the MoGe's Huggingface App



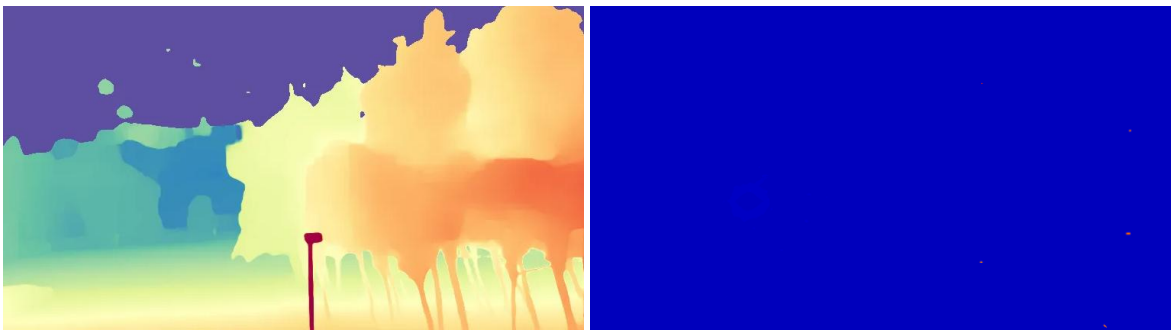
P3: Same Image run through Depth Anything v1 (also done by Marcel Ritter)

Since I can't play around with the parameters, I can at least compare the results of same inputs compared to other AI models, as seen in the images, one run through MoGe's algorithm and one through someone else's result using the Azure Kinects.

The input is an image of a relatively close-by environment, MoGe's result shows a clearly visible estimation of depth in which every distance is cleanly layered where else the Azure Kinect although shows a visible result has some troubles of determining where to put weight to the depth perception.



P4: Picture taken through Azure Kinect (by Marcel Ritter)



P5: Screenshot taken from the MoGe's Huggingface App

P6: Same Image run through Azure Kinect (also done by Marcel Ritter)

The input image shows a more open environment with multiple elements further away, the results show that the AI models some troubles with depth estimation when object in the image have a certain distance, although that is nothing compared the depth estimation from the Azure Kinect which didn't even manage to produce a visible result

Short Comparison

Although I have limited use with the Azure Kinect as well as the AI Model a couple of differences can be made out already:

Cost – Azure Kinect is going to set back the user a couple of hundred euros making it a rather expensive endeavour, MoGe is not costing anything and is free to use (except for a couple of braincells trying to get the thing to run)

Processing Result – From the two example I have tested, the Azure Kinect performed worse than MoGe, producing visibly worse results, making it the worse of the two.

Further tests that can be made - Speed/hardware comparison, I cannot with certainty say how much quicker any of the options are especially because MoGe is very much depended on the GPU and at least to my knowledge does not process real time videos, this is something that remains to be tested.

Overall, I would say hardware like the Azure Kinect although not the best but have their uses, as being able to process images in real time as you record gives them a certain edge over the ai models, but when it comes to processing single images the AI models are definitely a worthy alternative to the Azure Kinect.